Relatório de Estágio

Bruno Miguel Santos Martins

Engenharia Informática

jul | 2023

GUARDA
POLI
TÉCNICO

POLI TÉCNICO GUARDA

Escola Superior de Tecnologia e Gestão

HIPERINDEX BITSAPIENS

RELATÓRIO DE ESTÁGIO

PARA OBTENÇÃO DO GRAU DE LICENCIADO EM ENGENHARIA INFORMÁTICA

Bruno Miguel Santos Martins

Julho / 2023

Escola Superior de Tecnologia e Gestão

HIPERINDEX BITSAPIENS

RELATÓRIO DE ESTÁGIO

PARA OBTENÇÃO DO GRAU DE LICENCIADO EM ENGENHARIA INFORMÁTICA

Professor Orientador: José Fonseca

Bruno Miguel Santos Martins

Julho / 2023

Agradecimentos

Antes de tudo, gostaria de expressar minha profunda gratidão ao Instituto Politécnico da Guarda e aos professores do curso de Engenharia Informática pelo apoio contínuo e pelos preciosos conhecimentos que eles generosamente compartilharam comigo.

Gostaria de estender meus agradecimentos ao Professor José Fonseca, meu orientador, por sua constante disponibilidade, não apenas durante a execução do projeto, mas ao longo de toda a minha jornada acadêmica.

Quero agradecer à empresa BitSapiens por permitir que eu realizasse o meu estágio. Quero destacar a dedicação e apoio contínuo do meu supervisor, António Martins, assim como a disponibilidade constante do Hélder Gonçalves, que estiveram sempre dispostos a me auxiliar ao longo de toda a experiência de estágio.

Não podia deixar de agradecer à minha família e amigos por toda a força, apoio e motivação que me deram ao longo desta jornada.

Ficha de Identificação

Aluno

Nome: Bruno Miguel Santos Martins

Número: 1704756

Licenciatura: Engenharia Informática

Estabelecimento de Ensino

Instituto Politécnico da Guarda (IPG)

Escola Superior de Tecnologia e Gestão (ESTG)

Entidade Acolhedora do Estágio

Nome: Bitsapiens - Digital Business & Strategies Unipessoal, Lda

Morada: Rua Sra Abadia nº 325 / Fração J, 4750-063 - Barcelos

Contacto Telefónico: 919814035

Duração do Estágio: 06/07/2023 - 07/09/2023

Supervisor de Estágio

Nome: António Martins

Função: Chief Tecnhical Officer

Docente Orientador de Estágio

Nome: José Fonseca

Grau Académico: Doutor

Resumo

O presente documento tem como objetivo descrever o projeto desenvolvido em contexto de estágio integrado na Licenciatura de Engenharia Informática, realizado na empresa BitSapiens.

O projeto desenvolvido é uma aplicação web chamada HiperIndex para venda de questionários DISC. Os exames DISC avaliam os traços de Dominância, Influência, Estabilidade e Conformidade para proporcionar perceções sobre o comportamento e as preferências de uma pessoa em ambientes pessoais e profissionais. Esta aplicação está a ser desenvolvida por uma equipa de desenvolvimento constituída pelo estagiário e pelo *developer* Hélder Gonçalves da BitSapiens.

O principal objetivo desta aplicação é otimizar a eficiência e produtividade no ambiente de trabalho dos funcionários em empresas ou organizações. Essa melhoria é alcançada através da geração de uma avaliação quantitativa final para cada utilizador, o que, por sua vez, capacita os gestores a conduzirem análises mais aprofundadas do desempenho de cada funcionário.

A HiperIndex é composta por um *frontend*, que corresponde à parte da aplicação com a qual o utilizador irá interagir, desenvolvido em *React* e por um *backend*, que corresponde à parte da aplicação responsável por armazenar e manipular dados. O *frontend* foi desenvolvido por o estagiário e o *backend* foi desenvolvido pelo *developer* Hélder Gonçalves. Pela sua abrangência e complexidade, a HiperIndex não está inteiramente concluída, no entanto já permite realizar testes DISC, sendo este um dos objetivos principais ao fim do tempo de estágio.

Com o trabalho realizado, a BitSapiens fez uma oferta de trabalho ao estagiário, para que possa qual dar continuidade ao desenvolvimento desta e futuras aplicações.

Palavras-chave: aplicação web, questionários DISC, *React*, eficiência, avaliação quantitativa.

Abstract

This document aims to describe the project developed in the context of an internship integrated into the Computer Engineering Degree, carried out at the company BitSapiens.

The project developed is a web application called HiperIndex for selling DISC questionnaires. DISC exams assess the traits of Dominance, Influence, Stability and Conformity to provide insights into a person's behaviour and preferences in personal and professional environments. This application is being developed by a development team made up of the intern and the developer Hélder Gonçalves from BitSapiens.

The main objective of this application is to optimize efficiency and productivity in the working environment of employees in companies or organizations. This improvement is achieved by generating a final quantitative assessment for each user, which in turn empowers managers to conduct more in-depth analysis of each employee's performance.

HiperIndex is made up of a frontend, which corresponds to the part of the application with which the user will interact, developed in React, and a backend, which corresponds to the part of the application responsible for storing and manipulating data. The frontend was developed by the intern and the backend was developed by developer Hélder Gonçalves. Due to its scope and complexity, HiperIndex is not entirely completed. However, it already allows DISC tests to be carried out, which is one of the main objectives at the end of the internship period.

With the work done, BitSapiens made a job offer to the intern, so that he can continue to develop this and future applications.

Keywords: web application, DISC questionnaires, React, efficiency, quantitative assessment.

Índice

Índice de Figuras

Índice de Tabelas

Lista de Siglas

API	Application Programming Interface
CRUD	Create, Read, Update and Delete
CSS	Cascading Style Sheets
DISC	Dominance, Influence, Steadiness, and Conscientiousness
ESTG	Escola Superior de Tecnologia e Gestão
HTML	Hypertext Markup Language
IPG	Instituto Politécnico da Guarda
SQL	Structured Query Language
URL	Uniform Resource Locator

1 Introdução

O presente relatório descreve os tópicos desenvolvidos ao longo do estágio curricular realizado por Bruno Miguel Santos Martins, na Bitsapiens, no âmbito da Unidade Curricular Projeto de Informática do 3º ano da Licenciatura em Engenharia Informática, lecionada na Escola Superior de Tecnologia e Gestão (ESTG) do Instituto Politécnico da Guarda (IPG).

A produtividade é um aspeto crucial para o sucesso das empresas, e cabe aos gestores e diretores procurar formas de alcançar a máxima produtividade, eficiência dos seus colaboradores. Uma das abordagens para alcançar esse objetivo é por meio dos exames DISC.

Os exames DISC são uma metodologia para fazer a avaliação comportamental a fim de identificar os perfis dominantes de uma pessoa. O psicólogo americano William Moulton Marston teve a intenção de estabelecer uma maneira de decifrar as competências socio emocionais dos seres humanos, com objetivo de prever a tendência comportamental em determinados cenários. Por isso, segundo a teoria DISC, é possível decifrar quatro perfis predominantes: dominância(D), influência(I), estabilidade(S) e conformidade (C) (Sólides - DISC, 2023).

Tendo em conta esta necessidade de exames DISC por parte das empresas, foi criado este projeto que consiste na construção de uma aplicação web chamada HiperIndex. A HiperIndex irá permitir à Bitsapiens vender questionários DISC a empresas ou organizações para que possam otimizar os seus funcionários e assim ter uma maior produtividade e eficiência no trabalho e percebam de onde a podem extrair. O projeto descrito neste relatório consiste no desenvolvimento inicial da HiperIndex, que será continuada pelo estagiário, mas como funcionário da BitSapiens.

1.1 Motivação | Enquadramento

A minha motivação pessoal para este projeto é impulsionada pela minha curiosidade sobre a teoria DISC, que proporciona valiosas perceções sobre o comportamento humano, aliada ao meu entusiasmo pela tecnologia. O desenvolvimento da aplicação HiperIndex

representa uma oportunidade emocionante para automatizar e tornar mais acessível a utilização destes exames, contribuindo para a eficácia das empresas e organizações.

1.2 Caraterização sumária da Bitsapiens

A Bitsapiens é uma agência de estratégias de negócios digitais e Web3 que oferece de forma estratégica serviços personalizados no mundo digital - combinando Serviços de Consultoria e um Estúdio para gerenciar criações com o desenvolvimento de plataformas digitais e aplicativos. Esta é uma empresa relativamente recente no mercado, estando em atividade há cerca dois ano tendo sido fundada 2021.

O modelo operacional da Bitsapiens foi concebido para ser global, digital e multicultural. A empresa trabalha principalmente de forma remota, porém utiliza sua presença física em diversas localidades ao redor do mundo para desenvolver sua cultura, apoiar empreendedores e estabelecer relacionamentos (Bitsapiens, 2021).

1.3 Descrição do problema

Assim como outras plataformas que oferecem exames DISC, o projeto segue a mesma filosofia. Esses exames geralmente geram métricas que revelam comportamentos e habilidades dos colaboradores avaliados. Com base nessa análise, os gestores obtêm informações valiosas para identificar os pontos fortes, áreas de desenvolvimento e potencial de cada profissional dentro da empresa.

No entanto, o presente projeto diferencia-se ao oferecer uma abordagem inovadora nas respostas fornecidas aos utilizadores. Não apenas apresentamos respostas individuais, mas também uma visão coletiva para um grupo de utilizadores. Além disso, fornecemos uma resposta quantitativa final para cada utilizador, aprimorando a capacidade do gestor de realizar uma análise mais aprofundada sobre seus colaboradores como vemos na Figura 1.1 e Figura 1.2. Na Figura 1.1 observamos que o utilizador reactivo pois tem possui maior percentagem no eixo da serenidade e no eixo cumpridor como vemos na Figura 1.2.

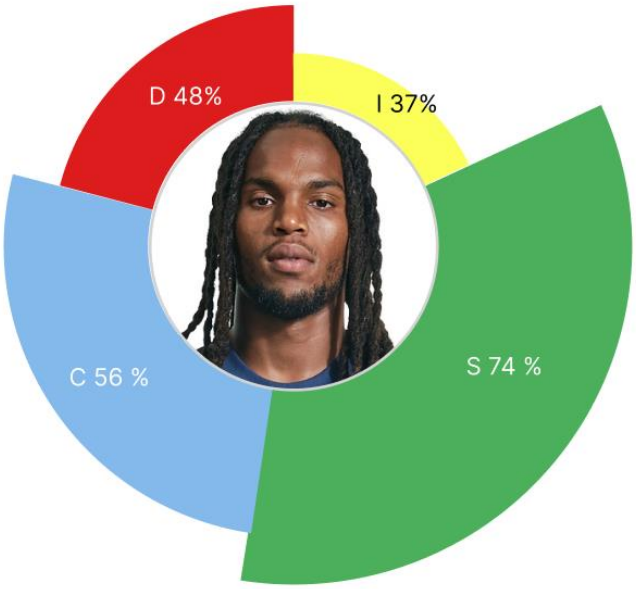


Figura 1.1 - Exemplo de mapa comportamental de um utilizador

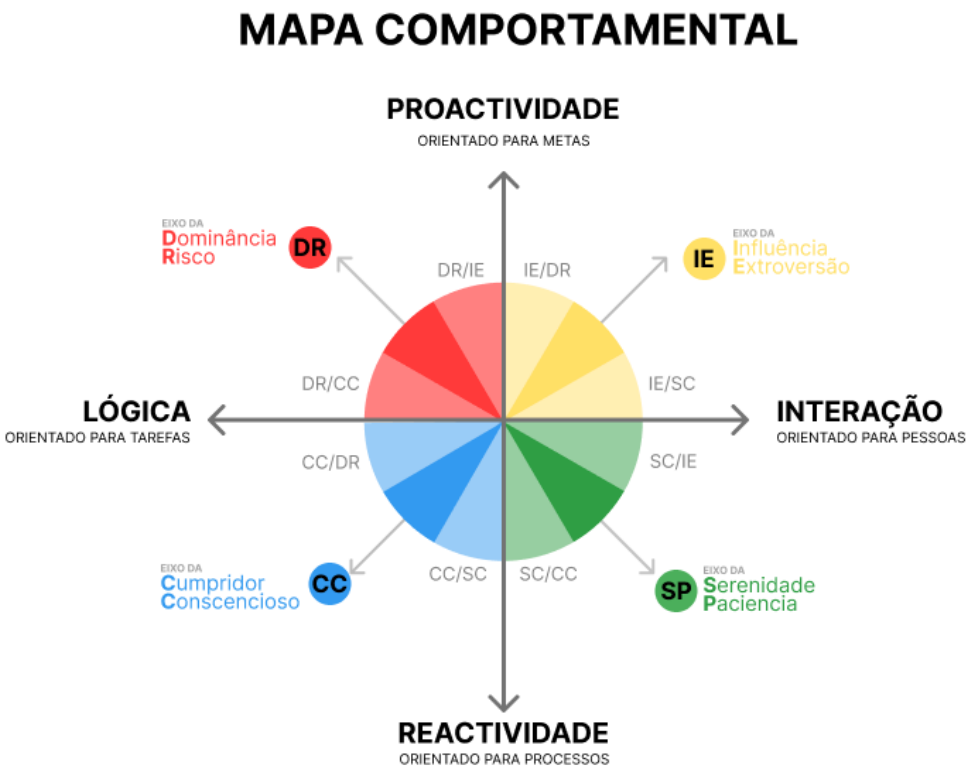


Figura 1.2 – Mapa Comportamental

Na Figura 1.1 observamos maior percentagem no eixo da serenidade e no eixo cumpridor, e assim através da Figura 1.2 podemos dizer que o utilizador é reactivo ou orientado a processos.

1.4 Objetivos

O objetivo principal deste projeto é a criação de uma aplicação web que permite fornecer aos utilizadores, a criação e realização de exames DISC para o aumento da produtividade e eficiência nas empresas.

Tendo em vista o proposto, o projeto deverá cumprir os seguintes objetivos:

- Análise das especificações do projeto e requisitos da plataforma web

Na análise de especificações do projeto proposto pela BitSapiens ao estagiário é a construção de aplicação web. Nos requisitos foram especificados como o projeto seria organizado e desenvolvido especificando como e o que iria fazer durante a duração do estágio.

- Desenvolver a interface web

O estagiário está responsável por realizar o *frontend* do projeto, sendo este a parte que é visualizada e interagida pelo utilizador. Durante esta fase foi importante o estagiário dispor todas as dúvidas possíveis durante a implementação ao *developer* Hélder.

- Documentar o desenvolvimento

Foi essencial o estagiário organizar e documentar, com a ajuda do *developer* Hélder, todo o desenvolvimento feito na aplicação.

- Execução de testes

Durante o desenvolvimento do projeto, era importante testar se as funcionalidades que foram implementadas estavam a funcionar através de metodologias próprias para tal.

1.5 Estrutura do relatório

O relatório está dividido em oito capítulos. O segundo capítulo é composto pela análise do estado de arte no que diz respeito aos exames DISC. No terceiro capítulo está descrita a metodologia de trabalho que foi aplicada ao projeto. No quarto capítulo corresponde à análise de requisitos onde foi feito o planeamento da aplicação. O quinto capítulo são abordadas as tecnologias utilizadas. Seguindo-se o sexto capítulo que contém o

desenvolvimento da aplicação. No sétimo capítulo faz-se uma breve descrição dos testes exemplificando um dos testes efetuados à aplicação. Por fim, são apresentadas no oitavo capítulo as conclusões do trabalho.

2 Estado de Arte

O estado de arte é uma das partes mais importantes de todo o trabalho, uma vez que, é uma referência ao estado atual de conhecimento sobre um determinado tópico que está a ser objeto de estudo.

Neste caso em específico, este capítulo aborda os exames DISC.

Como foi dito anteriormente, os exames DISC são uma metodologia para fazer a avaliação comportamental a fim de identificar os 4 perfis dominantes de uma pessoa, sendo eles: dominância(D), influência(I), estabilidade(S) e conformidade (C)

Dominância: Destacam se neste tipo de comportamento aqueles que geralmente são bastante proativos, assumindo a liderança em várias situações para impulsionar decisões e contribuir para o sucesso. Eles têm uma abordagem pragmática na busca por resultados satisfatórios e, por conseguinte, têm uma inclinação competitiva. Abordá-los de maneira eficaz muitas vezes envolve fornecer *feedbacks* concisos e diretos.

Influência: A capacidade de persuasão é uma das suas características mais proeminentes. Portanto, ao fornecer *feedback* sobre o desempenho deles, é aconselhável destacar como a avaliação pode servir como um guia para aprimorar ainda mais suas habilidades persuasivas.

Estabilidade: A pessoa é capaz de gerenciar suas emoções e se manter estável diante de mudanças de cenário, mantendo uma postura constante mesmo em face de adversidades. Ao fornecer *feedback* a esses colaboradores, é aconselhável expressar a avaliação por meio de exemplos específicos para uma compreensão mais clara e concreta.

Conformidade: Agem de maneira precisa, aderindo estritamente às regras e diretrizes estabelecidas pela empresa. Portanto, ao dar o *feedback*, é recomendável contextualizar a situação usando as próprias normas e políticas da instituição como base para explicação.

Os questionários DISC são uma ferramenta essencial para o setor de Recursos Humanos por exemplo, onde é impactado positivamente em diversas áreas, incluindo o recrutamento e seleção, desenvolvimento dos colaboradores, plano de carreira e *feedback* de desempenho (Sólides - DISC, 2023).

O teste DISC é aplicado por meio de questionários e formulários. Com as respostas, é possível gerar métricas que indicam comportamentos e competências dos colaboradores avaliados. Em geral, os testes são de escolha múltipla, e pedem respostas diretas sobre temas como preferências pessoais, opinião e as atividades que mais gosta de desempenhar. O resultado vai apontar a característica dominante do indivíduo (Sólides - DISC, 2023).

Os resultados dos testes DISC são determinados por meio de cálculos e algoritmos específicos, sendo que cada aplicação possui o seu próprio método. Não há uma fórmula única de cálculo; ela varia conforme o algoritmo utilizado. Geralmente, as respostas, que são múltipla escolha ou envolvem ordenação, são associadas a uma determinada "pontuação". Essas pontuações são então utilizadas nos cálculos, resultando na identificação do tipo de perfil do usuário.

“Hoje, há uma pressão crescente sobre as empresas para recolherem essas perspetivas sobre os seus trabalhadores, à medida que os executivos se debatem com decisões dispendiosas sobre a necessidade de trabalho presencial no escritório ou mesmo a manutenção do espaço de escritório. No mínimo, os testes de personalidade podem dar às empresas o vocabulário para falar sobre como os seus trabalhadores gostam de socializar: se desejam brincadeiras mais frescas ou temem a festa de fim de ano.” (The $2 Billion Question of Who You Are at Work, 2023).

Como podemos ver no parágrafo anterior, este tipo de exames é importantíssimo para uma boa eficiência e organização numa empresa. Daqui nasce uma oportunidade de negócio que dá início ao projeto.

3 Metodologia

Neste capítulo é descrita a metodologia que foi adotada para a realização do projeto. A metodologia refere se a um conjunto de processos, técnicas, procedimentos e diretrizes organizadas e estruturadas para abordar e resolver problemas ou realizar tarefas específicas de maneira sistemática e eficiente.

A metodologia utilizada para o desenvolvimento de software foi uma versão aproximada da metodologia Agile pois ao longo do meu percurso enquanto estagiário foi necessária uma adaptação às mudanças nos requisitos de forma a executar e reforçar o planeamento de todo o projeto. Esta metodologia foi sugerida pelo supervisor sendo que é a metodologia que usam mais para os seus projetos.

3.1 Metodologia Agile

Este modelo assenta numa abordagem interativa. Caracteriza-se por dividir cada projeto em pequenas partes que devem ser completadas semanalmente ou em curtos períodos de tempo. Assim, torna-se mais simples compreender prioridades e introduzir mudanças no desenvolvimento do projeto, caso seja necessário.

Trata-se, no fundo, de um modelo de trabalho em equipa, pensado para levar a cabo projetos organizados segundo os objetivos e necessidades dos clientes. (Jasmin Software - Metodologia Agile, 2023).

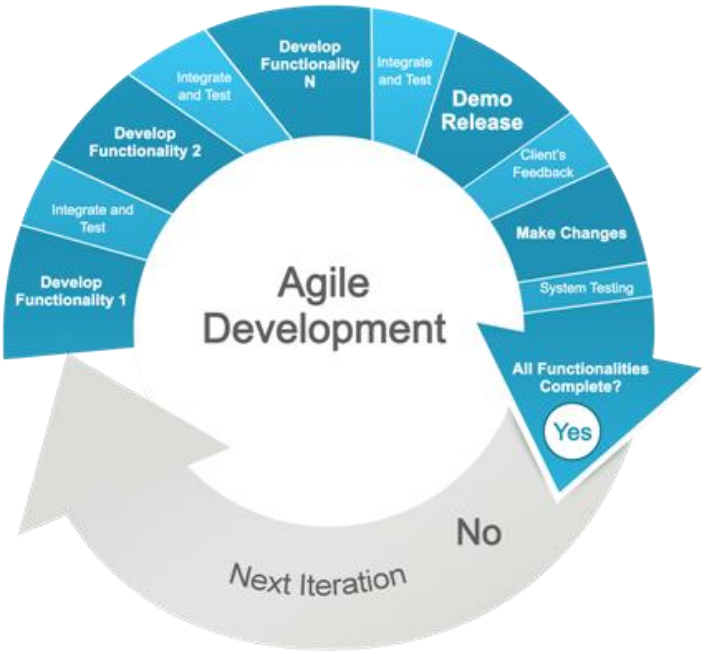


Figura 3.1 - Metodologia Agile

Como podemos observar na Figura 3.1 a metodologia Agile é feita através do realização, integração e testes de n funcionalidades numa primeira fase. Em seguida, é feito um protótipo para o perceber *feedback* do cliente. Por fim, o *feedback* é usado para informar e incorporar nos requisitos da próxima iteração. Este é um elemento fundamental do processo iterativo, pois possibilita melhorias contínuas e ajustes com base no *feedback* do mundo real.

3.2 Integração da Metodologia Agile

O projeto HiperIndex foi desenvolvido pelo estagiário, o *developer* Hélder Gonçalves e pelo supervisor de estágio António Martins. No início do projeto, realizamos uma reunião para discutir como seria feito o desenvolvimento e quais funcionalidades seriam implementadas. Após a análise dos requisitos e especificações do projeto, começamos a trabalhar no desenvolvimento da aplicação web.

Durante a implementação, havia um diálogo diário com o *developer* Hélder Gonçalves para discutir as funcionalidades que estavam a ser e iriam ser implementadas. Todos os dias, eram designadas tarefas para o estagiário desenvolver, tal como, eram apontadas dúvidas, erros e melhorias para as próximas etapas do projeto. No final do dia, o estagiário relatava o que havia sido feito ao *developer* e ao supervisor.

4 Análise de Requisitos

A análise de requisitos é uma parte essencial da gestão de projetos, pois envolve a busca de informações cruciais e necessárias para entender o que o utilizador precisa, resolver um problema e atingir os objetivos desejados. Além disso, também visa compreender as expectativas do utilizador em relação a um determinado produto (Analise de requisitos - InfoEscola, s.d.).

4.1 Atores e respetivos casos de uso

Casos de uso representa uma possível utilização do sistema por um ator, que pode ser uma pessoa, dispositivo físico, mecanismo ou subsistema que interage com o sistema alvo, utilizando algum de seus serviços (Nakagawa, 2017).

A Tabela 4.1 contem os atores da aplicação e os seus respetivos casos de uso.

Tabela 4.1 - Casos de Uso e Atores

Atores	Casos de Uso
Admin	• Gerir a plataforma • CRUD Questionários • CRUD Utilizadores • CRUD Índices
Clientes - Administradores	• Acesso aos resultados dos questionários • Acompanharem o índice • Aplicar e usar questionários DISC • Usar *templates* de índices para avaliar os clientes
Staff Cliente	• Criar dados para índice
Avaliados	• Responderem a questionários

Destes atores apenas existe um na versão em que o projeto se encontra implementado até à data, que é o *Admin*. Este consegue criar, ler, atualizar e apagar (CRUD) questionários, Clientes, Utilizadores e Índices, para além de conseguir gerir a aplicação web, visto que é o ator com maior poder na aplicação. O cliente administrador é o que utilizador que dentro da empresa têm mais permissões. O cliente staff é um utilizador que tem permissões limitadas pelo cliente administrador. O avaliado é um funcionário ou membro da empresa ou organização que apenas responde aos questionários.

4.2 Diagrama de casos de uso

Como podemos observar na Figura 4.1 estão representados os atores com os seus respetivos casos de uso. Alguns dos casos de uso são mostrados fora da fronteira porque ainda não foram implementados no projeto.

Figura 4.1 - Diagrama de caso de uso (Fonte: Elaboração própria)

Como podemos observar na Figura 4.1, o ator *Admin* é o aquele que consegue realizar todos os casos de uso. Todos casos de uso são incluídos pelo caso de uso Login pois é necessário o Login para fazer qualquer tipo de operação na aplicação. Os casos de uso que se encontram fora da fronteira serão funcionalidades a implementar no futuro.

4.3 Descrição de casos de uso

Nesta secção é descrito em formato de tabela os Casos de uso, na descrição é usado o seguinte *template*:

- Nome: nome do caso de uso;
- Descrição: descrição do caso de uso. Deve ser uma frase simples que descreve a funcionalidade do caso de uso;
- Pré-Condição: condição inicial necessária para que o caso de uso se realize com sucesso;
- Caminho Principal: sequência de passos que o ator deve realizar para executar com sucesso a ação descrita pelo Caso de Uso;
- Caminho Secundário: sequência de passos quando há uma falha numa determinada ação do caminho principal;
- Pós-condições: condição em que o sistema se encontra após a finalização do caso de uso em questão

4.3.1 Responder questionário

O Caso de uso descrito na Tabela 4.2 refere-se ao conjunto de passos que o Utilizador tem que realizar para responder a um questionário DISC.

Tabela 4.2 – Descrição de caso de uso - Responder Questionário

Nome	**Responder questionário**
Descrição	Estes casos de uso têm como objetivo o utilizador responder a um questionário DISC.
Pré-Condição	Login Válido. Existir questionário/s DISC.
Caminho Principal	1. O ator acede á página de Login da aplicação. 2. O sistema valida o Login e redireciona o ator para a página “Painel” da aplicação 3. O ator abre a sidebar selecionando o botão na parte esquerda da navbar seleciona o botão “Exames” na Sidebar. 4. O sistema abre a página “Exames” da aplicação 5. O ator seleciona o botão de “Criar exame”

	6. O sistema abre um card com as informações de criação do questionário DISC. 7. O ator cria o exame e começa a responder ao exame.
Caminho Secundário	3. a) O ator seleciona um exame através de um card na página "Painel" 5. a) O ator seleciona o botão "Concluir" de um exame que está por concluir.
Pós-Condição	

4.4 Descrição da aplicação

Numa primeira fase, foi planeado de acordo com a especificações do projeto e requisitos da plataforma web que o estagiário iria seguir o protótipo que se encontra na aplicação Figma para auxiliá-lo na construção do *frontend* do projeto. Estão dispostos 6 ecrãs de protótipo da aplicação que se encontra no Figma, como vemos na Figura 4.2, em que o estagiário para ter auxílio para o desenvolvimento e implementação da aplicação web.

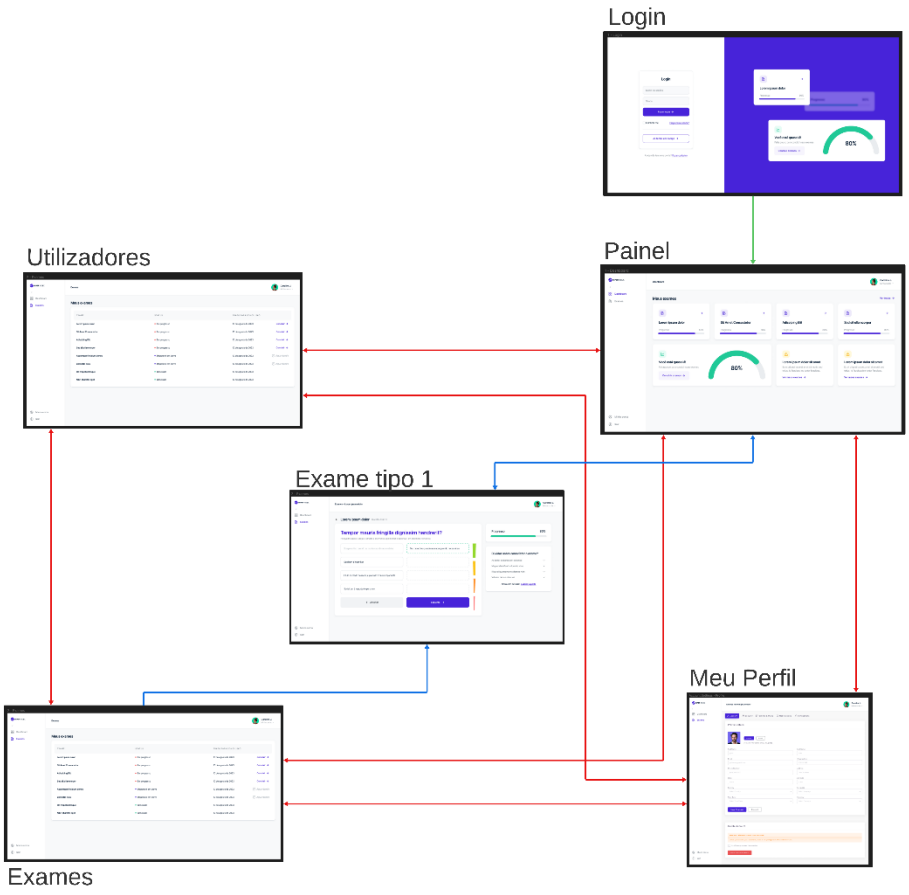


Figura 4.2 - Diagrama de navegação da aplicação (Fonte: Elaboração própria)

Como vemos na Figura 4.2, será necessário fazer registo ou *login*. Depois do registo ou *login* feito, os utilizadores serão reencaminhados para a página principal da aplicação chamada Painel. Nesta página e em todas as outras do protótipo tem dois elementos muito importantes chamados de *navbar* e a *sidebar*. É a partir da *sidebar* que vai ser possível navegar entre as páginas como vemos na Figura 4.2.

No caso para perceber melhor a navegação na aplicação, na Figura 4.2 as ligações estão com cores diferentes. A ligação a verde significa que é única porque é logo depois de o utilizador ter feito o login na aplicação. As ligações a vermelho significam que dá para navegar de página para página através da *sidebar* e *navbar*. A ligações a azul significam que aquelas páginas são a única maneira de acessar á página de exame através de botões específicos para tal.

Na página "Exames" consegue-se visualizar através de uma tabela os exames que existem, estando estes feitos ou por fazer. Se o *Admin* quiser criar exame ele terá uma opção para tal. No caso de o avaliado continuar um que já exista, este depois de selecionar

o respetivo botão na tabela irá ser redirecionado para uma página onde é feito o exame DISC que o próprio pretende continuar.

O exame DISC apresenta uma página onde podemos ver 4 frases onde o utilizador irá ordená-las consoante a relevância que têm para o utilizador. Também é mostrado uma área que mostra o progresso do utilizador no exame, e outra que mostra as dúvidas que normalmente poderão ter. Depois de ter concluído é mostrado ao utilizador uma página que o exame foi concluído com sucesso.

A página “Utilizadores” só pode ser visualizada pelos *Admin*. Nesta página é possível visualizar uma tabela com todos os utilizadores que existem mostrando apenas informação básica dos mesmos, onde também é possível criar utilizadores a partir desta página.

Na página “Meu Perfil” é possível editar e guardar informação sobre o próprio, como também pode fazer a eliminação permanente da conta.

5 Tecnologias

Neste capítulo serão apresentadas as tecnologias utilizadas para o desenvolvimento da aplicação.

5.1 React

O *React* é uma biblioteca *front-end JavaScript* de código aberto com foco em criar interfaces de utilizador em páginas web (Gackenheimer, 2015).

5.1.1 JavaScript

JavaScript é uma linguagem de programação interpretada estruturada, de *script* em alto nível com tipagem dinâmica fraca e multiparadigma (protótipos, orientado a objeto, imperativo e funcional). Juntamente com HTML e CSS, o *JavaScript* é uma das três principais tecnologias da World Wide Web. *JavaScript* permite páginas da Web interativas e, portanto, é uma parte essencial dos aplicativos da web. A grande maioria dos sites usa, e todos os principais navegadores têm um mecanismo *JavaScript* dedicado para executá-lo (Wikipédia - JavaScript, s.d.).

5.2 Node.js

Node.js é um software de código aberto, multiplataforma, baseado no interpretador V8 do *Google* e que permite a execução de códigos JavaScript fora de um navegador web. A principal característica do Node.js é sua arquitetura assíncrona e orientada por eventos (Wikipédia - Node.js, s.d.).

5.3 HTML

HTML (abreviação para a expressão inglesa HyperText Markup Language, que significa: "Linguagem de Marcação de Hipertexto") é uma linguagem de marcação utilizada na construção de páginas na Web. Documentos HTML podem ser interpretados por navegadores (Wikipédia - HTML, s.d.).

5.4 CSS/

Cascading Style Sheets é um mecanismo para adicionar estilos (cores, fontes, espaçamento, etc.) a uma página web, aplicado diretamente nas tags HTML ou ficar contido dentro das *tags* <*style*>. Também é possível, adicionar estilos adicionando um *link* para um arquivo CSS que contém os estilos. Assim, quando se quiser alterar a aparência dos documentos vinculados a este arquivo CSS, basta modificá-lo (Wikipédia - Cascading Style Sheets, s.d.).

5.5 Visual Studio Code

O *Visual Studio Code* é um editor de código-fonte desenvolvido pela *Microsoft* para *Windows, Linux e macOS*. Ele inclui suporte para depuração, controle de versionamento *Git* incorporado, realce de sintaxe, complementação inteligente de código, *snippets* e refatoração de código. Ele é customizável, permitindo que os utilizadores possam mudar o tema do editor, teclas de atalho e preferências. Por estas razões, acabou por ser o editor de código ideal para o projeto a ser desenvolvido (Wikipédia - Visual Studio Code, s.d.).

5.6 Figma

Figma é um editor gráfico de vetor e prototipagem de projetos de *design* baseado principalmente no navegador web, com ferramentas *offline* adicionais para aplicações *desktop* para *GNU/Linux*, *macOS* e *Windows*. O *Figma* é um software focado no desenvolvimento de sistemas de *design* gráfico, prototipagem de interface gráfica de utilizador e desenvolvimento de UI/UX (experiência da interface com o utilizador), permitindo também o desenvolvimento colaborativo em tempo real com outros utilizadores remotamente. Esta ferramenta serviu como prototipagem de interface do que será a nossa aplicação web (Wikipédia - Figma, s.d.).

5.7 ClickUp

ClickUp é uma plataforma de produtividade completa que funciona como um local ideal para as equipas se reunirem, debaterem ideias, planejarem e colaborarem em tudo, desde documentos de processos até *designs* de produtos. Embora depender de uma única ferramenta para tantos casos de uso possa parecer um exagero, o *ClickUp* foi criado para

isso. Esta tecnologia serviu para nos organizarmos e percebermos o que cada um precisava tinha de fazer ou já tinha feito no projeto (Luz.vc - ClickUp, s.d.).

6 Implementação

Como foi inicialmente referido, o estagiário trabalhou no projeto em conjunto com o *developer* Hélder Gonçalves, tendo cada uma das partes do projeto sido distribuída por cada um dos membros. Assim, o estagiário trabalhou na implementação do *frontend* e o *developer* Hélder Gonçalves trabalhou na implementação do *backend*.

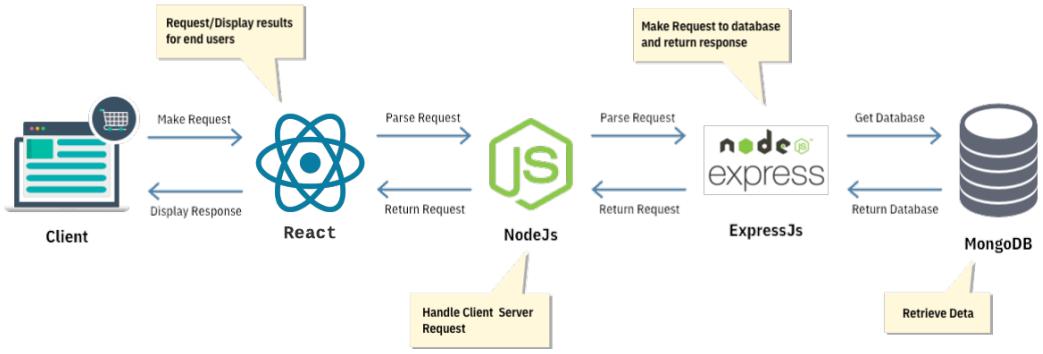


Figura 6.1 - Diagrama de arquitetura (Fonte: Elaboração própria)

Na Figura 6.1 mostra o diagrama de arquitetura da aplicação que consiste em três principais componentes interconectados. No lado do cliente, a interface do utilizador é construída com React. Do lado do servidor, a API Node.js gerência solicitações HTTP, direcionando-as para controladores e serviços específicos que implementam a lógica de negócios. Esses serviços interagem com o MongoDB, um banco de dados NoSQL, para armazenar e recuperar dados.

Nos tópicos abaixo serão explicadas algumas funcionalidades do desenvolvimento do estagiário neste projeto de um modo mais detalhado.

6.1 Implementação da página Login

A primeira página implementada foi o *login*, uma vez que, a primeira interação do utilizador com a aplicação dá-se nesta página.

Existe dois formatos para os quais as páginas irão ser feitas: ecrãs de grande dimensão como computadores e ecrãs de menor dimensão como telemóveis, *tablets* entre outros.

Para o estagiário saber o que teve fazer nesta página, ele utilizou a ferramenta *Figma* que dispõe de todo o design que era pretendido para o projeto como vemos na Figura 6.2

Assim, para esta página o estagiário teve de ter em atenção as *templates* da aplicação *Figma* para a ecrãs de grande dimensão e para ecrãs de pequena dimensão como vemos na Figura 6.3 e na Figura 6.4 respetivamente.

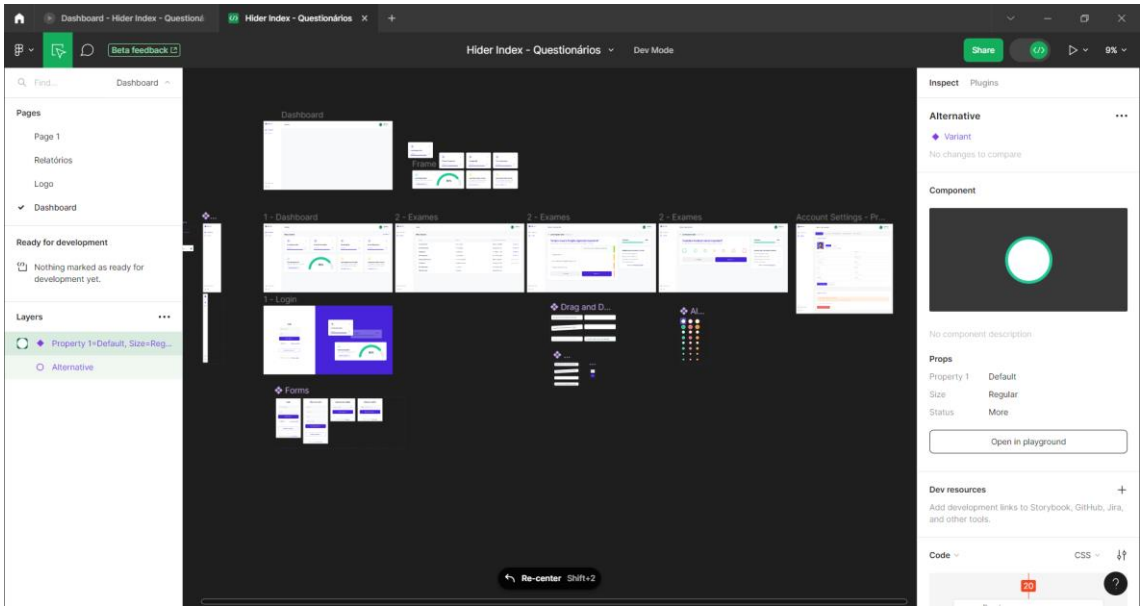


Figura 6.2 - Aplicação Figma

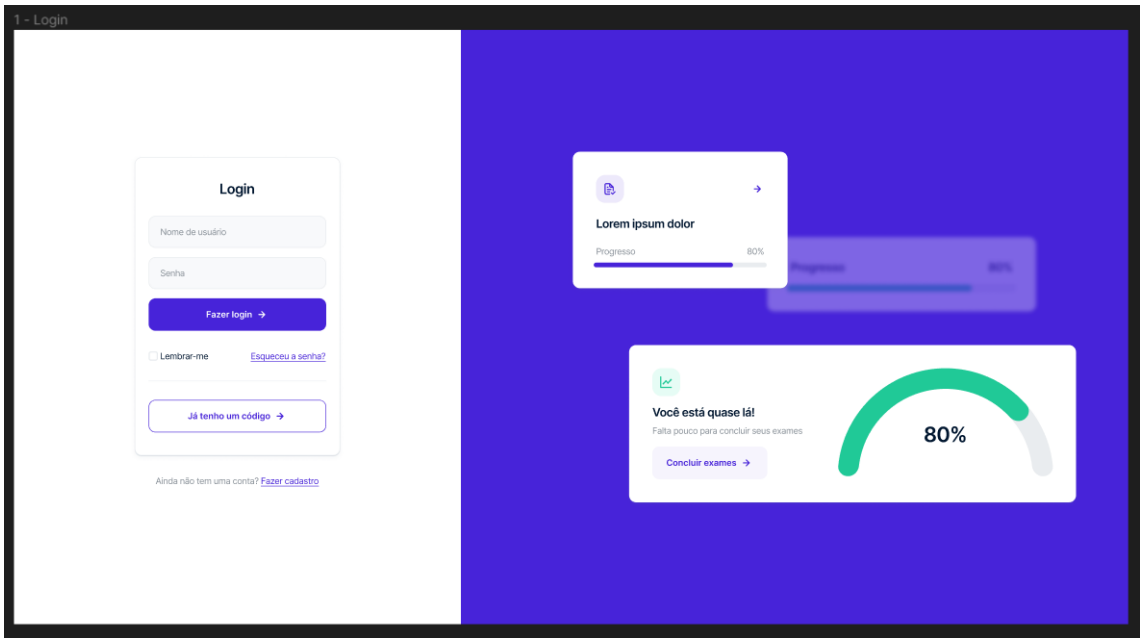


Figura 6.3 - Página Login no Figma para ecrãs de grandes dimensões

Figura 6.4 - Página Login no Figma para ecrãs de pequenas dimensões

Neste projeto, também foi respeitado e usado as boas normas para fazer código onde as pastas e ficheiros são divididos o máximo possível para que o estagiário e o *developer* Hélder consigam trabalhar de uma forma mais eficiente e organizada como vemos na Figura 6.5,.

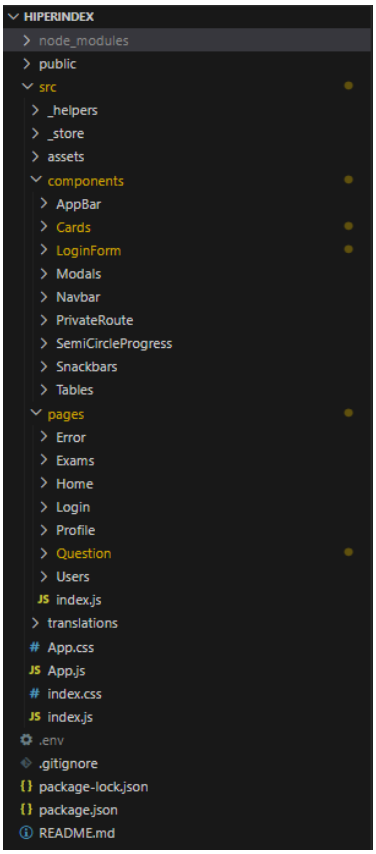


Figura 6.5 - Estrutura da aplicação

Na Figura 6.5 têm uma pasta "*pages*" onde contém todas as páginas do projeto e na pasta "*components*" contém componentes que foram usados nessas mesmas páginas. No caso da página *Login*, os componentes usados foram os *forms* na pasta "*LoginForm*", que servem para abrir a página principal do projeto de formas diferentes. Um dos *forms* usados para a página Login foi o que se apresenta na Figura 6.6.

CodeForm.jsx 3

src > components > LoginForm > CodeForm.jsx > ...

```jsx
import styles from "./styles.module.css";
import * as HiIcons from "react-icons/hi";
import { useForm } from "react-hook-form";

const CodeForm = ({ data, handleChange, onSubmit, goToLogin, authError }) => {
  const { register, handleSubmit, formState } = useForm();
  const { errors, isSubmitting } = formState;

  return (
    <div className={styles.login_form_container}>
      <form className={styles.form_container} onSubmit={handleSubmit(onSubmit)}>
        <h4>Acessar com código</h4>
        <div className={styles.form_container_labels}>
          <input
            type="text"
            placeholder="Digite o código"
            name="code"
            onChange={handleChange}
            value={data.code}
            required
            className={styles.input}
          />
          {authError && (
            <div className={styles.error_msg}>{authError.message}</div>
          )}
          <button
            type="submit"
            className={styles.purple_btn}
            disabled={isSubmitting}
          >
            Acessar exame
            <HiIcons.HiArrowRight />
          </button>
        </div>
      </form>
      <div className={styles.cadastro_container}>
        Já tem uma conta?{" "}
        <a onClick={goToLogin} href="#">
          Fazer login
        </a>
      </div>
    </div>
  );
};

export default CodeForm;
```

Figura 6.6 - Form de código de acesso usado para Login

Na Figura 6.6 podemos ver na linha 5 são passadas propriedades como: *data*, *handleChange*, *onSubmit*, *goToLogin e authError*. Entre as linhas 13 e 22 no campo de entrada é para o utilizador inserir um código. A função *handleChange* é chamada sempre que o campo de entrada muda e o valor atual do campo de entrada é armazenado em *data.code*. Se houver um erro de autenticação (*authError*), será exibida a mensagem de erro como vemos nas linhas 23 a 25. O botão *submit* fica desabilitado enquanto o formulário está a ser enviado. Quando o formulário é enviado, ele chama a função *onSubmit*.

Este formulário, tal como os outros, são então importados para a página *Login* como vemos na Figura 6.7, tornando assim o código muito mais organizado e eficiente como já foi dito anteriormente.

Login.jsx 1 ×

src > pages > Login > Login.jsx > Login > handleAccessCode

```
1   import { useState, useEffect } from "react";
2   import styles from "./styles.module.css";
3   import { loginBanner } from "../../assets/images";
4   import { logoSvg } from "../../assets/svg";
5   import { CodeForm, SignUpForm, LoginForm } from "../../components/LoginForm";
6   import { useSelector, useDispatch } from "react-redux";
7   import { history } from "../../_helpers";
8   import { authActions } from "../../_store";
9
10  const Login = () => {
11    const [data, setData] = useState({ email: "", password: "", code: "" });
12    const [showLoginFields, setShowLoginFields] = useState(true);
13    const [showSignUpFields, setShowSignUpFields] = useState(false);
14    const dispatch = useDispatch();
15    const authUser = useSelector((x) => x.auth.user);
16    const authError = useSelector((x) => x.auth.error);
17
18    useEffect(() => {
19      // redirect to home if already logged in
20      if (authUser) history.navigate("/");
21    }, []);
22
23    const handleChange = ({ currentTarget: input }) => {
24      setData({ ...data, [input.name]: input.value });
25    };
26
27    const onSubmit = async (e) => {
28      return dispatch(
29        authActions.login({ email: data.email, password: data.password })
30      );
31    };
32
33    const goToCode = () => {
34      setShowLoginFields(false);
35      setShowSignUpFields(false);
36    };
37
38    const goToLogin = () => {
39      setShowLoginFields(true);
40      setShowSignUpFields(false);
41    };
42
43    const goToSignUp = () => {
44      setShowSignUpFields(true);
45      setShowLoginFields(false);
46    };
47
48    const handleAccessCode = (e) => {
49      return dispatch(authActions.loginViaAccessCode({ accessCode: data.code }));
50    };
```

Figura 6.7 – Código página Login (Parte 1)

Como vemos na linha 5 da Figura 6.7 os *imports* dos *forms* que vão ser essenciais para a página. Também foram criadas algumas variáveis, entre elas "*data*" e "*authError*" que foram usadas também no *form* do código de acesso como foi visto na Figura 6.6.

Na Figura 6.8 podemos então perceber os componentes *SignUpForm*, *LoginForm* e *CodeForm* recebem várias propriedades, incluindo o estado dos dados, a função

handleChange, a função de envio apropriada e as funções para navegar para outros formulários.

Figura 6.8 - Código página Login (Parte 2)

Assim chegamos a este resultado que vemos na Figura 6.9 para ecrã de grandes dimensões e a Figura 6.10 para ecrãs de pequenas dimensões.

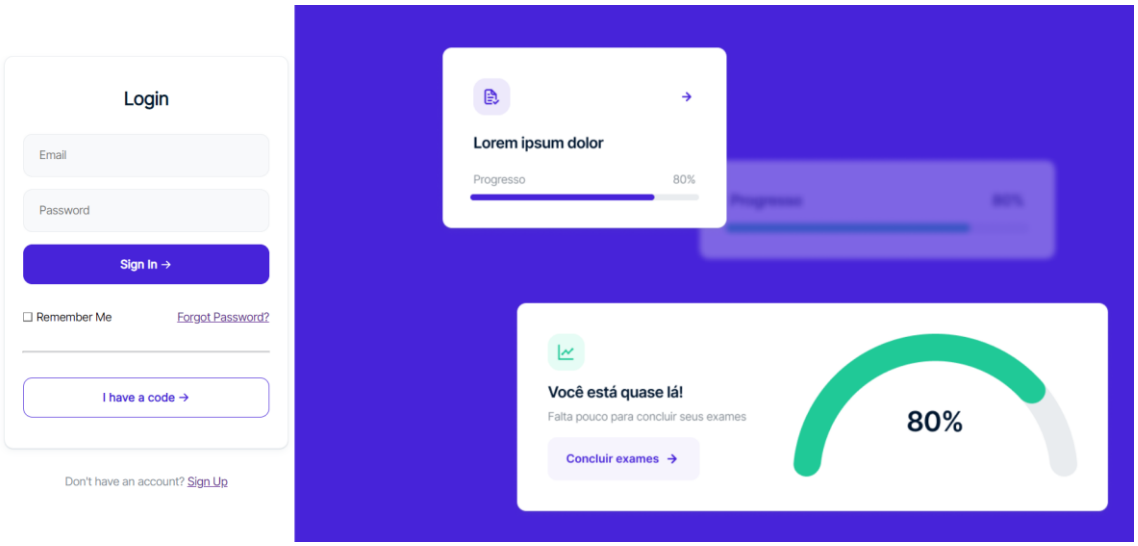


Figura 6.9 - Página Login para ecrãs de grandes dimensões

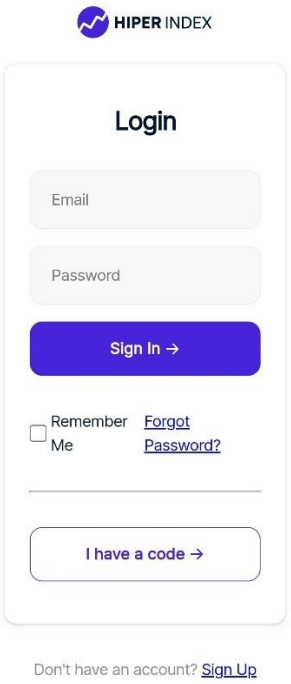


Figura 6.10 - Página Login para ecrãs de pequenas dimensões

6.2 Implementação da Navbar e Sidebar

Uma vez que entramos na aplicação esta teria de conter dois elementos importantes em todas as páginas web que são a *Navbar* e a *Sidebar*. A *Navbar* e a *Sidebar* são uma secção interface gráfica de utilizador destinada a auxiliar o acesso à informação. Na Figura 6.11 podemos observar as variáveis criadas para a criação da *Navbar* e *Sidebar*.

index.jsx 1 ×

src > components > Navbar > index.jsx > Navbar

```
22
23  const Navbar = ({ sidebar, setSidebar }) => {
24    const [t, i18n] = useTranslation("common");
25    const showSidebar = () => setSidebar(!sidebar);
26    const [anchorEl, setAnchorEl] = React.useState(null);
27    const user = useSelector((x) => x.auth.user || null);
28    const dispatch = useDispatch();
29    const open = Boolean(anchorEl);
30    const handleClick = (event) => {
31      setAnchorEl(event.currentTarget);
32    };
33    const handleClose = () => {
34      setAnchorEl(null);
35    };
36
37    if (!user) {
38      return null;
39    }
40
41    const handleLogout = () => dispatch(authActions.logout());
42
43    const logout = () => dispatch(authActions.logout());
44
45    const SidebarData = [
46      {
47        title: t("sidebar.dashboard"),
48        path: "/",
49        icon: <BsIcons.BsGrid />,
50        cName: styles.navtext,
51      },
52      {
53        title: t("sidebar.exams"),
54        path: "/exams",
55        icon: <AiIcons.AiOutlineFileDone />,
56        cName: styles.navtext,
57      },
58      {
59        title: t("sidebar.users"),
60        path: "/users",
61        icon: <AiIcons.AiOutlineUser />,
62        cName: styles.navtext,
63      },
64      {
65        title: t("sidebar.profile"),
66        path: "/profile",
67        icon: <PiIcons.PiUserCircleLight />,
68        cName: styles.navtext3,
69      },
70      {
71        title: t("sidebar.logout"),
72        path: "/",
73        icon: <PiIcons.PiSignOut />,
74        onClick: logout,
75        cName: styles.navtext2,
76      },
77    ];
78
```

Figura 6.11 - Código do elemento Navbar e Sidebar para ecrãs de pequena dimensão

Na Figura 6.11 nas linhas 23 a 35 são definidas várias variáveis (*showSidebar*, *handleClick*, *handleClose*, *handleLogout*, *logout*) para lidar com a alternância da barra lateral, abertura e fechamento de menu e *logout* do utilizador. Na linha 45 a 77 contêm a variável "*SideBarData*" que contém objetos que representam os *links* de navegação. Cada objeto possui um título, um caminho, um ícone, um "cName" (nome da classe para estilo) e uma função "*onClick*" que é apenas para o botão "Sair".

Na Figura 6.12 podemos ver o código feito para a página compatível com dispositivos *mobile*.

Figura 6.12 - Código do elemento Navbar e Sidebar para ecrãs de pequena dimensão (Parte 2)

Na Figura 6.12 da linha 84 a 88 inclui um componente *Link*, que está num ícone, que quando clicado, aciona a função *showSidebar* para alternar a visibilidade da *Sidebar*. Na linha 90 a 104 inclui um componente *IconButton* que, quando clicado, aciona a função *handleClick*. O botão contém um componente *avatar* e o primeiro nome do utilizador ou "Convidado" se o utilizador não estiver com um utilizador registado na aplicação.

Da linha 107 a 127 ele renderiza um menu de navegação que contém uma imagem do logotipo e uma lista de *links* de navegação.

Na linha 112 a 125 temos então os *links* de navegação são gerados a mapear a matriz *SidebarData*.

Vemos então na Figura 6.13 e na Figura 6.14 como ficou a *Navbar* e *Sidebar* na página Painel da perspetiva de um ecrã de pequenas dimensões.

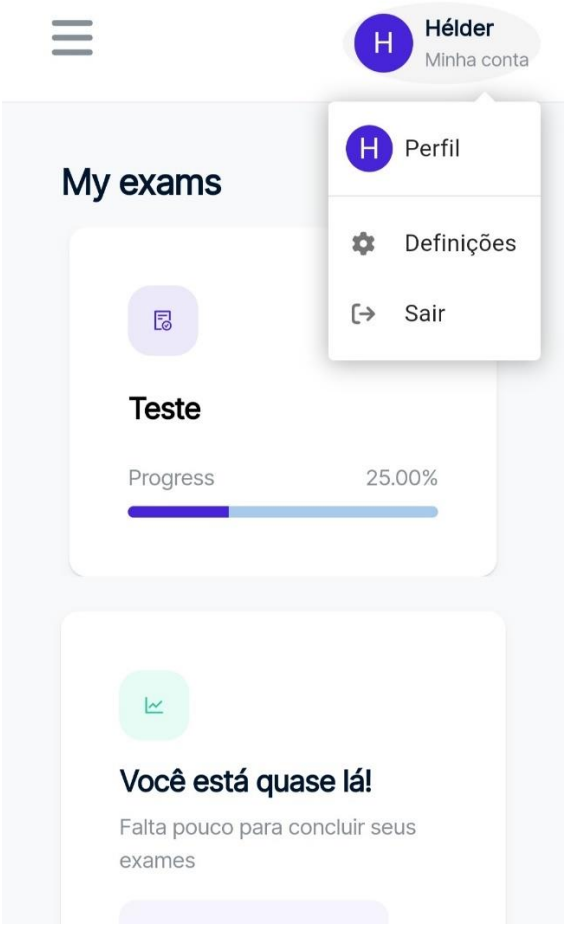


Figura 6.13 - Navbar num ecrã de pequenas dimensões

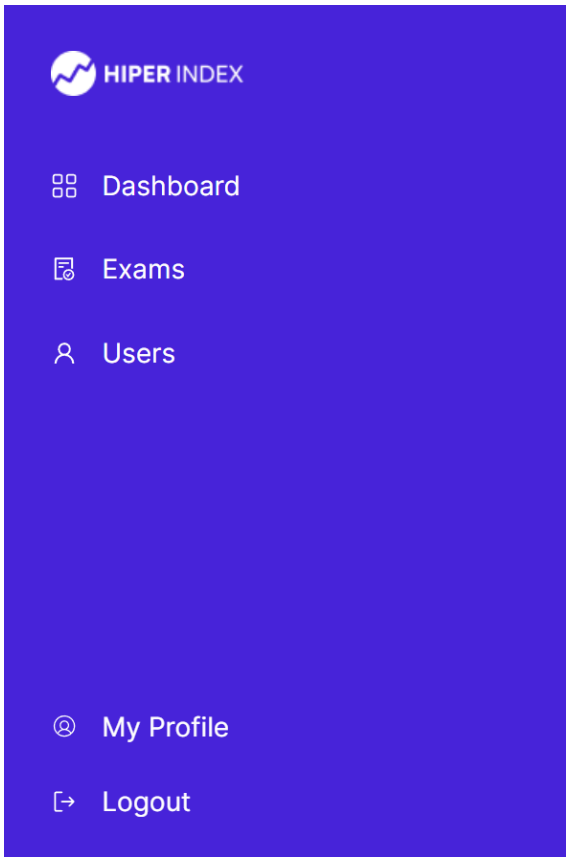


Figura 6.14 - SideBar num ecrã de pequenas dimensões

A aplicação já usada para ecrãs de maior dimensão apresenta os elementos com uma aparência diferente. Como vemos na Figura 6.15 o código utilizado para fazer a *Navbar* e *Sidebar* para ecrãs de grandes dimensões.

index.jsx 1

src > components > Navbar > index.jsx > Navbar

```jsx
<div className={styles.normalview}>
  <IconContext.Provider value={{ color: "#868E96" }}>
    <div className={sidebar ? styles.navbar : styles.navbaractive}>
      <Link to="#" className={styles.menuBars}>
        <FaIcons.FaBars onClick={showSidebar} />
      </Link>
      <div className={styles.accountcontainer}>
        <Avatar sx={{ bgcolor: "#4723d9" }}>
          {user ? user.firstName[0] : "G"}
        </Avatar>
        <div className={styles.user}>
          {user ? user.firstName : "Guest"}
          <div className={styles.myaccount}>{t("navbar.account")}</div>
        </div>
      </div>
    </div>
    <nav className={sidebar ? styles.navmenuactive : styles.navmenu}>
      <ul className={styles.navmenuitems} onClick={showSidebar}>
        <li>
          <img src={logo} alt="Logo" className={styles.logo} />
        </li>
        {SidebarData.map((item, index) => {
          return (
            <li key={index} className={item.cName} onClick={item.onClick}>
              {item.path ? (
                <Link to={item.path}>
                  {item.icon}
                  <span>{item.title}</span>
                </Link>
              ) : (
                <item.onClick />
              )}
            </li>
          );
        })}
      </ul>
    </nav>
  </IconContext.Provider>
</div>
```

Figura 6.15- Código do elemento Navbar e Sidebar

O código que vemos na Figura 6.15 é muito idêntico ao código que vemos na Figura 6.12 pois a única coisa que muda são as propriedades CSS nas classes usadas que vemos na Figura 6.15.

Podemos então ver como ficou a *Navbar* e *Sidebar* na página Painel da perspetiva de um ecrã de grandes dimensões como num computador como podemos ver na Figura 6.16 e Figura 6.17.

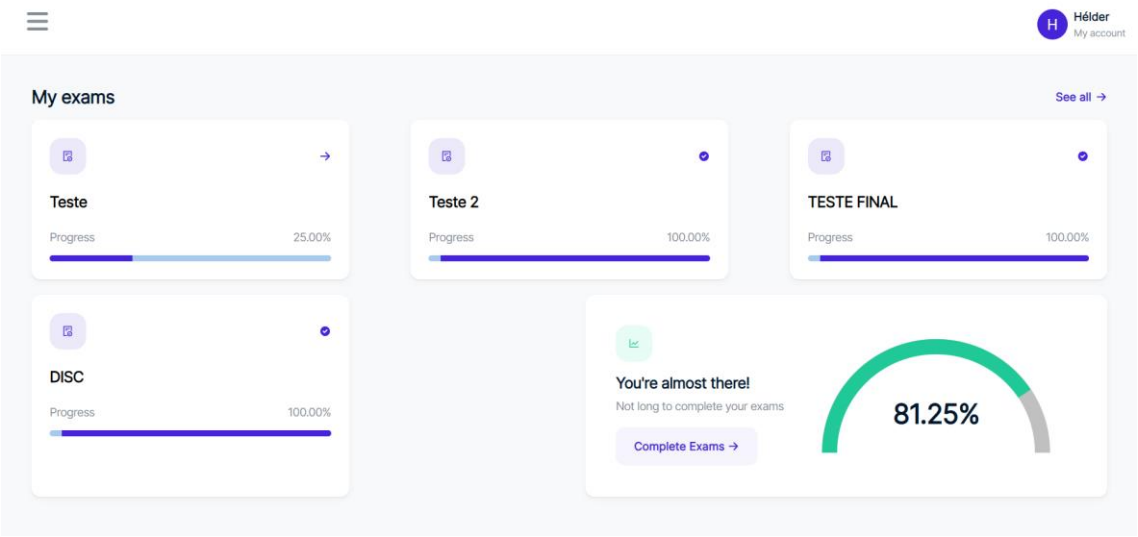


Figura 6.16 - NavBar num ecrã de grandes dimensões

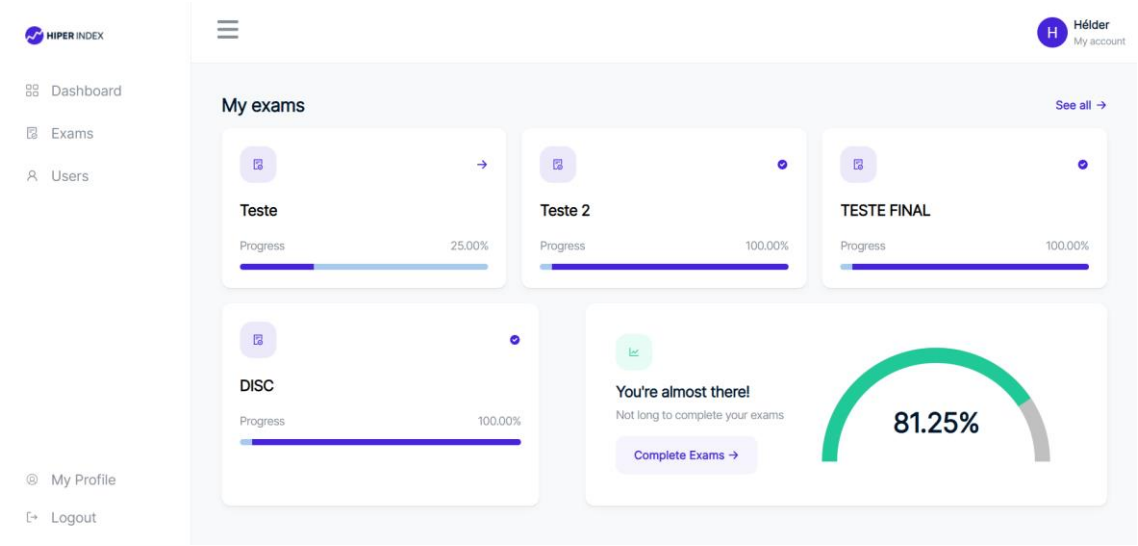


Figura 6.17 - Navbar e SideBar num ecrã de grandes dimensões

6.3 Implementação da página Painel

A segunda página a ser implementada foi a página Painel. O estagiário através da aplicação *Figma* vai usar as *templates* que se pretende para aquela página, tanto para versão de ecrãs de pequenas e grandes dimensões, como se vê na Figura 6.18 e na Figura 6.19.

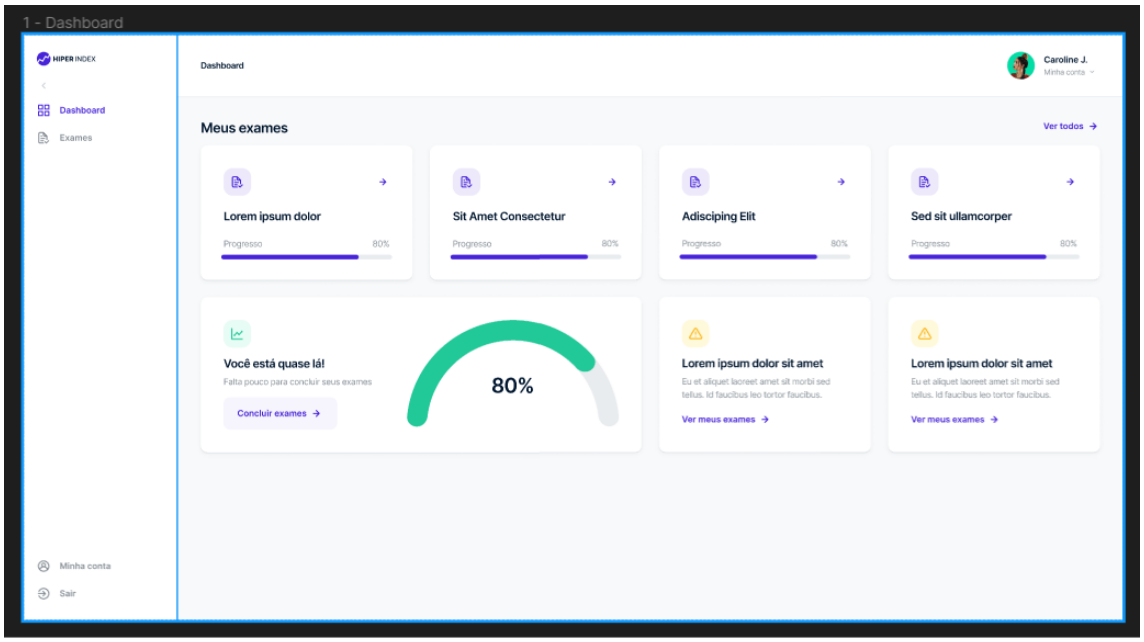


Figura 6.18 - Página Painel no Figma para grandes dimensões

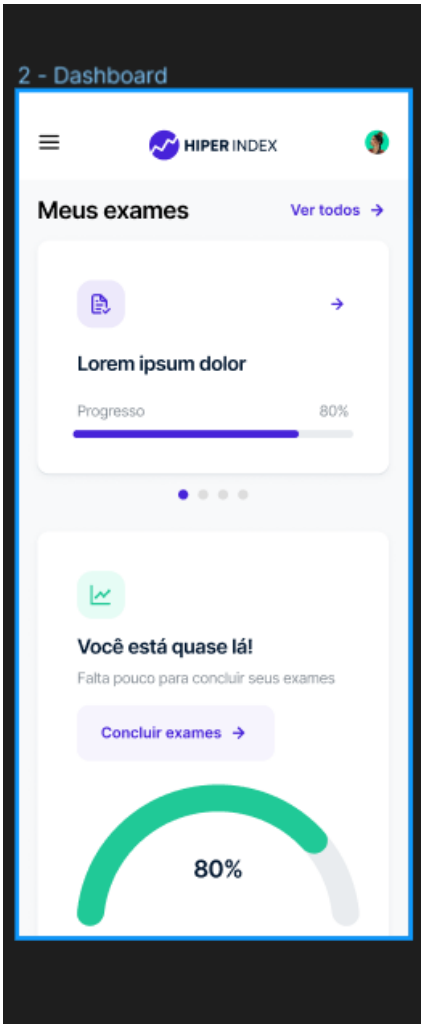


Figura 6.19- Página Painel no Figma para pequenas dimensões

Como observamos na Figura 6.18 o estagiário colocou um elemento na página chamado de *Card*, que é visto no centro da página. O *Card* consiste em vários elementos da interface do utilizador, incluindo cabeçalhos, subcabeçalhos, imagens, multimédia, botões de ação e divisórias.

Assim pode-se observar na Figura 6.20 e na Figura 6.21 código feito para a página Painel para ecrãs de grandes dimensões.

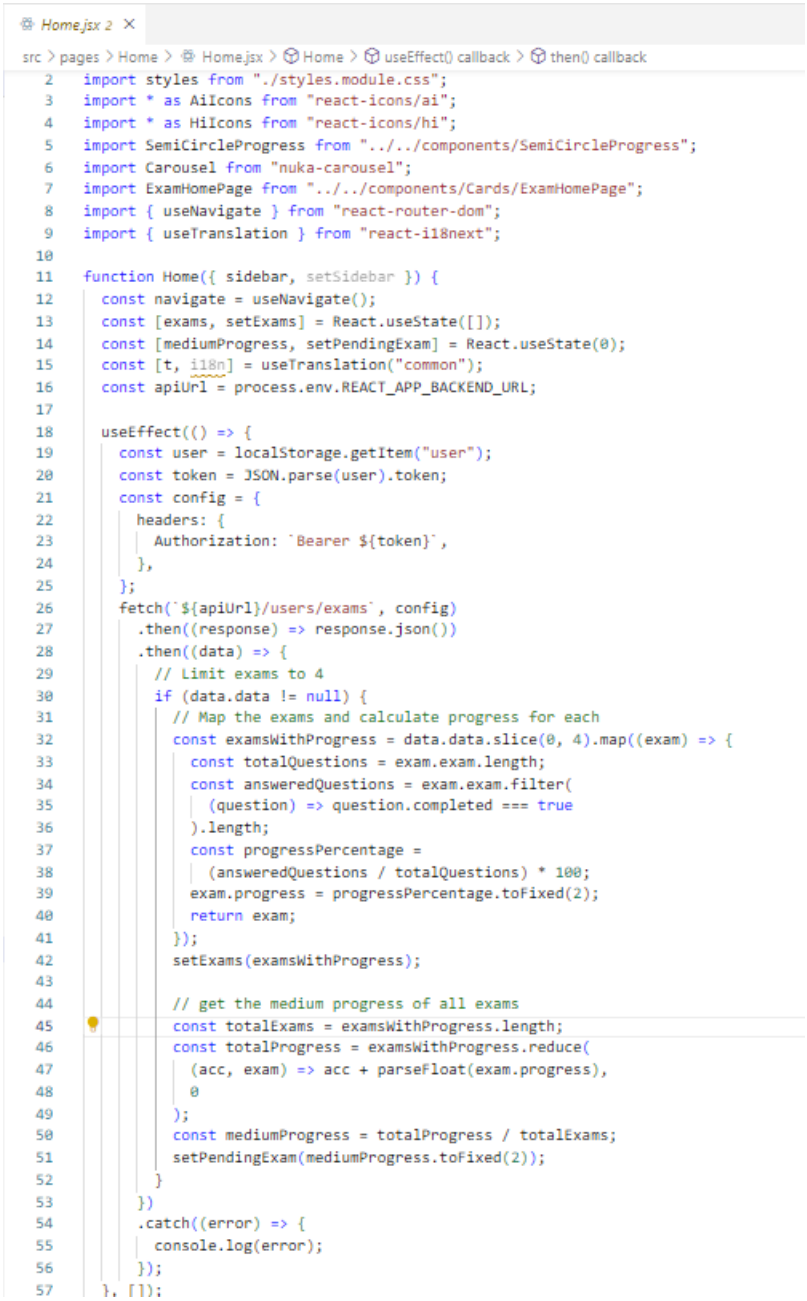


Figura 6.20 - Código página Painel (Parte 1)

Figura 6.21 - Código página Painel (Parte 2)

Na Figura 6.20 da linha 12 a 16 temos o *hook useNavigate* que serve para navegar para diferentes rotas, o *hook useState* serve para gerenciar o estado local para exames, o *hook useTranslation* irá ser para traduzir o texto e por fim uma constante *apiUrl* que contém a URL base da API.

Da linha18 a 57 é definido um *hook useEffect* que é executado uma vez quando o componente é montado. Dentro do *hook useEffect*, ele recupera o *token* do utilizador do armazenamento local e configura os cabeçalhos para a solicitação da API. É enviado uma solicitação GET ao *endpoint* "/users/exams" da API para procurar os exames do utilizador. Se a resposta da API não for nula, ela limitará o número de exames a 4, calculará o progresso de cada exame e definirá o estado dos exames como uma série de exames com seu progresso. Calcula-se também o progresso médio de todos os exames e

define o estado *mediumProgress* para esse valor. Se houver um erro com a solicitação da API, ele irá registar esse erro na consola.

Na Figura 6.21 na linha 59 é definido uma função *goToExams* que navega para a rota "/*exams*" quando chamada. Na linha 74 a 79 é mapeado a variável de estado dos exames e renderiza um componente *ExamHomePage* para cada exame. O componente *ExamHomePage* recebe propriedades para o título, progresso e ID do exame, sendo assim este um tipo de *cards* que é feito.

O outro *card*, que podemos ver da linha 82 a 106, renderiza um componente *SemiCircleProgress* que exibe o progresso médio de todos os exames. O componente *SemiCircleProgress* recebe adereços para o raio do círculo, largura do da barra de progresso, percentagem do progresso, cor do progresso, para além também de ter o título, descrição e um botão.

Na Figura 6.22 podemos ver o código feito para a página Painel para ecrãs de pequenas dimensões.

Home.jsx 2

src > pages > Home > Home.jsx > Home

```
107         <div className={styles.cardsmobile}>
108           <Carousel withoutControls={true} cellSpacing={12}>
109             {exams.map((exam) => (
110               <ExamHomePage
111                 title={exam.name}
112                 progress={exam.progress || 0}
113                 examId={exam._id}
114               />
115             ))}
116           </Carousel>
117           <div className={styles.card2}>
118             <div className={styles.card2_container1}>
119               <div className={styles.rectanglegreen}>
120                 <AiIcons.AiOutlineLineChart />
121               </div>
122               <div className={styles.container_almosthere}>
123                 <h5>{t("home.almost_there")}</h5>
124                 {t("home.almost_there_description")}
125               </div>
126               <button className={styles.finishexams_btn} onClick={goToExams}>
127                 {t("home.button_finish")} <HiIcons.HiArrowRight />
128               </button>
129             </div>
130             <div className={styles.card2_container2}>
131               <SemiCircleProgress
132                 circleRadius={150}
133                 progressWidth={20}
134                 percentage={Number(mediumProgress)}
135                 progressColor={"#20C997"}
136               >
137                 <h2 className={styles.text_percentage}>{mediumProgress}%</h2>
138               </SemiCircleProgress>
139             </div>
140           </div>
141         </div>
142       </div>
143     );
144   }
145
146   export default Home;
147
```

Figura 6.22 - Código página Painel (Parte 3)

Na Figura 6.22 da linha 107 a 115 temos outro tipo de *card* onde é utilizado o elemento carrossel para mostrar a informação ao utilizador. O carrossel é uma apresentação de *slides* para percorrer uma série de conteúdo, construída com transformações CSS, 3D e um pouco de *JavaScript*. Vemos que é usado a variável de estado dos exames e renderiza um componente *ExamHomePage* para cada exame como tínhamos visto anteriormente. Para além disso, o seguinte *card* também é bastante similar ao que é visto da linha 82 a 106 na Figura 6.21.

Assim com isto, é apresentado o seguinte resultado visto na Figura 6.23 para ecrãs de grandes dimensões e na Figura 6.24 e na Figura 6.25 para ecrãs de pequenas dimensões.

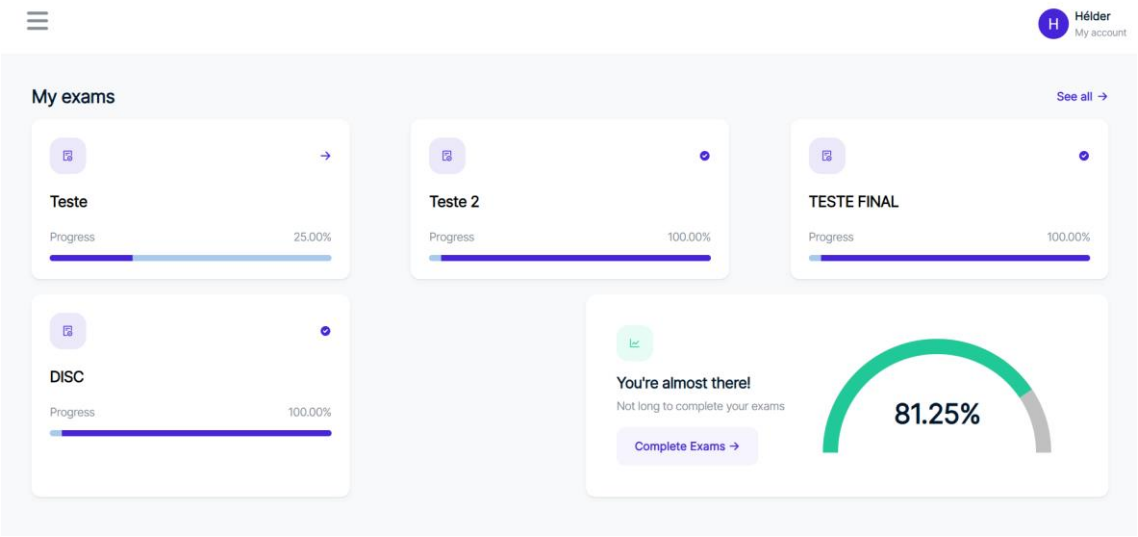


Figura 6.23 - Página Painel num ecrã de grandes dimensões

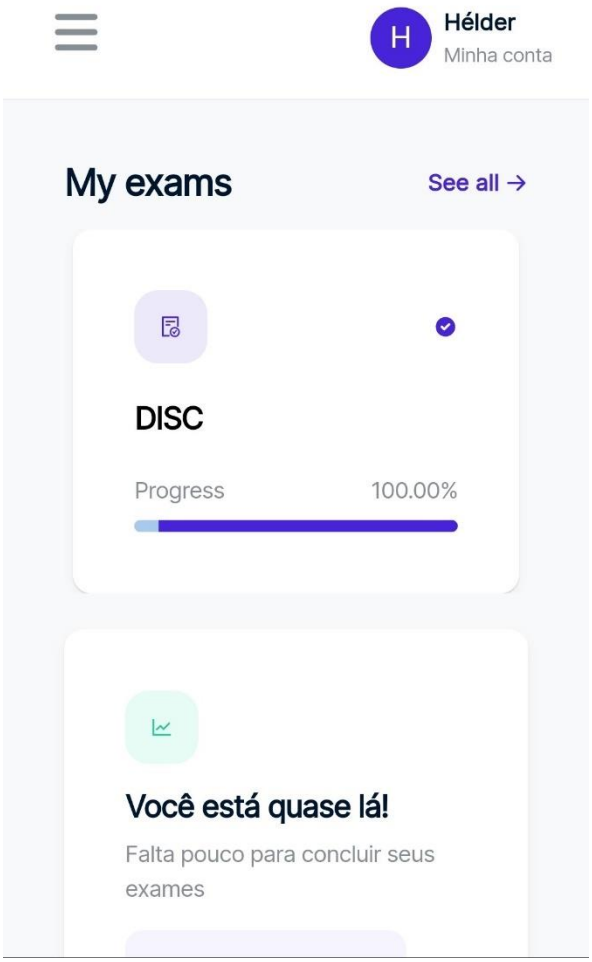


Figura 6.24 - Página Painel num ecrã de pequenas dimensões (Parte 1)

Figura 6.25 - Página Painel num ecrã de pequenas dimensões (Parte 2)

6.4 Implementação da página Exames

A terceira página a ser implementada foi a página Exames. O estagiário através da aplicação *Figma* vai usar as *templates* que se pretende para aquela página, tanto para versão de ecrãs de pequenas e grandes dimensões, como se vê na Figura 6.26 e na Figura 6.27.

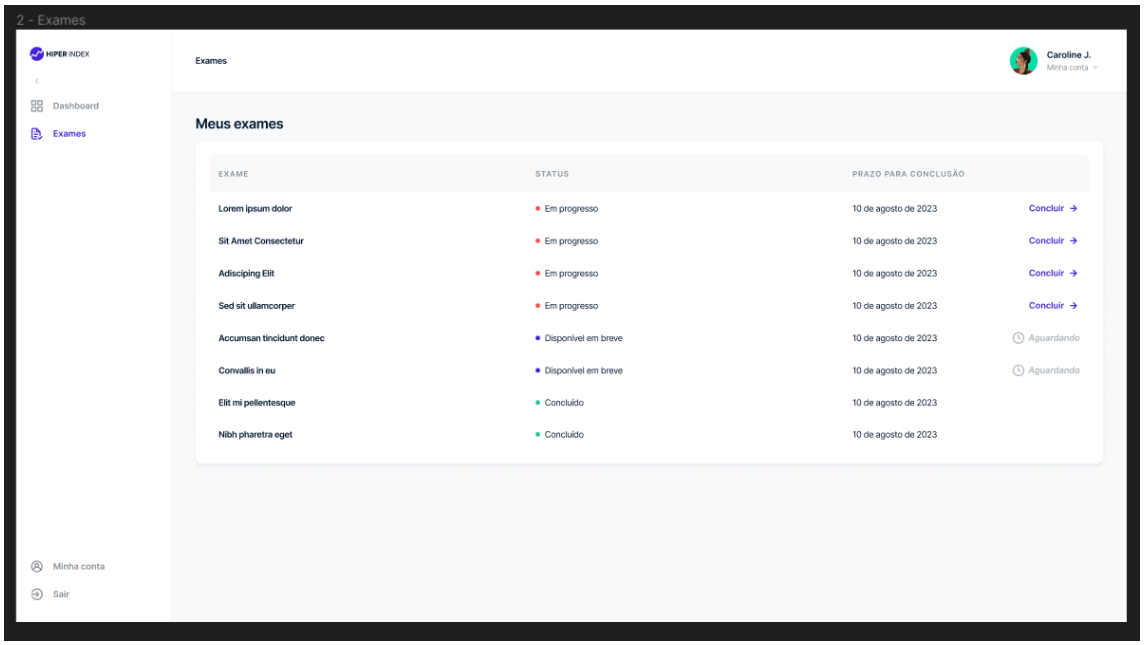


Figura 6.26 - Página Exames no Figma para grandes dimensões

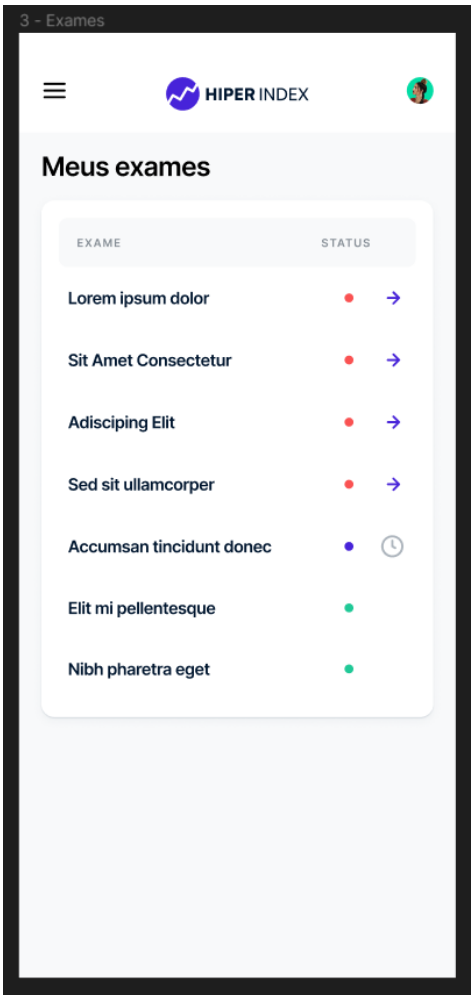


Figura 6.27 - Página Painel no Figma para pequenas dimensões

Na Figura 6.26 e na Figura 6.27 podemos observar que existe um elemento que mostra os exames do utilizador, no qual lhe chamamos de tabela. O código criado para este elemento foi criado num ficheiro na pasta "*componentes/Tables/ExamsTables*" para deixar o código mais eficiente e organizado. Vemos então o código da tabela de exames na Figura 6.28.

index.jsx 1 ×

src > components > Tables > ExamsTable > index.jsx > ...

```
13  function ExamsTable({ header, data, goToExam }) {
14    const [t, i18n] = useTranslation("common");
15    return (
16      <TableContainer
17        component={Paper}
18        sx={{
19          borderRadius: "12px",
20          backgroundColor: "#fff",
21          boxShadow:
22            "0px 1px 1px 0px rgba(118, 135, 154, 0.24), 0px 3px 8px 0px rgba(118, 135, 154, 0.08)",
23        }}
24      >
25        <Table sx={{ minWidth: 650 }} aria-label="simple table">
26          <TableHead
27            sx={{
28              backgroundColor: "var(--gray-gray-1, #F8F9FA)",
29              borderRadius: "12px",
30            }}
31          >
32            <TableRow sx={{ borderRadius: "12px", border: "none" }}>
33              {header.map((columnHeader, index) => (
34                <TableCell
35                  key={index}
36                  sx={{
37                    fontWeight: "bold",
38                    color: "var(--gray-gray-6, #868E96)",
39                    fontSize: "14px",
40                  }}
41                >
42                  {columnHeader.title}
43                </TableCell>
44              ))}
45            </TableRow>
46          </TableHead>
47          <TableBody>
48            {data.map((row, rowIndex) => (
49              <TableRow
50                sx={{
51                  "&:hover": {
52                    backgroundColor: "var(--gray-gray-1, #F8F9FA)",
53                  },
54                }}
55                key={row._id}
56              >
57                <TableCell
58                  key={`name-${row._id}`}
59                  sx={{
60                    color: "var(--Dark, #00162F)",
61                    fontFamily: "Inter",
62                    fontSize: "15px",
63                    fontStyle: "normal",
64                    fontWeight: 600,
65                    lineHeight: "150%", // Corrected to use string value
66                    letterSpacing: "-0.45px",
67                  }}
68                >
```

Figura 6.28 - Código da tabela da página Exames (Parte 1)

Na Figura 6.28 na linha 13 são três propriedades: "*header*", "*data*" e "*goToExam*". É usado um componente "*TableContainer*" que contém uma tabela.

Na linha 16 a 68 vemos como este componente funciona. A tabela possui uma propriedade "*TableHead*", "*TableBody*", "*TableRow*" e um "*TableCell*". O "*TableHead*" contém um "*TableRow*" que mapeia o "*header*" para gerar componentes e o "*TableCell*" para cada cabeçalho de coluna. Cada "*TableRow*" no "*TableBody*" contém um "*TableCell*" que exibe o nome do exame. A linha muda a sua cor de fundo quando passa o rato sobre a mesma. A propriedade "*header*" é uma matriz de objetos, cada um com uma propriedade "*title*". A propriedade *data* seja uma a matriz de objetos de exame, cada um com uma propriedade "*id*" e "*name*". Os componentes "*TableContainer", "Table", "TableHead", "TableRow"* e *"TableCell*" são todos modificados usando a propriedade "*sx*".

Por fim na Figura 6.29, vemos que o estado do exame pode ser um dos 3 seguintes:

- Pendente;
- Em progresso;
- Completo;

index.jsx 1

src > components > Tables > ExamsTable > index.jsx > ExamsTable > data.map() callback

```
                <TableCell
                  key={`status-${row._id}`}
                  sx={{
                    color: "var(--Dark, #00162F)",
                    gap: "8px",
                    fontSize: "15px",
                    fontStyle: "normal",
                    fontWeight: 400,
                    lineHeight: "150%", // Corrected to use string value
                    letterSpacing: "-0.45px",
                  }}
                >
                  {row.status === "PENDING" && (
                    <IconContext.Provider
                      value={{ color: "#4723D9", size: "8px" }}
                    >
                      <div
                        style={{
                          display: "flex",
                          alignItems: "center",
                          gap: "8px",
                          fontSize: "15px",
                        }}
                      >
                        <BsIcons.BsFillCircleFill />
                        {t("exams.pending")}
                      </div>
                    </IconContext.Provider>
                  )}

                  {row.status === "COMPLETED" && (
                    <IconContext.Provider
                      value={{ color: "#51CF66", size: "8px" }}
                    >
                      <div
                        style={{
                          display: "flex",
                          alignItems: "center",
                          gap: "8px",
                          fontSize: "15px",
                        }}
                      >
                        <BsIcons.BsFillCircleFill />
                        {t("exams.completed")}
                      </div>
                    </IconContext.Provider>
                  )}

                  {row.status === "IN_PROGRESS" && (
                    <IconContext.Provider
                      value={{ color: "#FA5252", size: "8px" }}
                    >
                      <div
                        style={{
                          display: "flex",
                          alignItems: "center",
```

Figura 6.29 - Código da tabela da página Exames (Parte 2)

O estado do exame vai influenciar como é mostrado a informação nas colunas de estado e ações como vemos na Figura 6.29.

Na Figura 6.30 temos então o código da página Exame onde vai ser importado o ficheiro com o código tabela visto anteriormente na Figura 6.28 e na Figura 6.29.

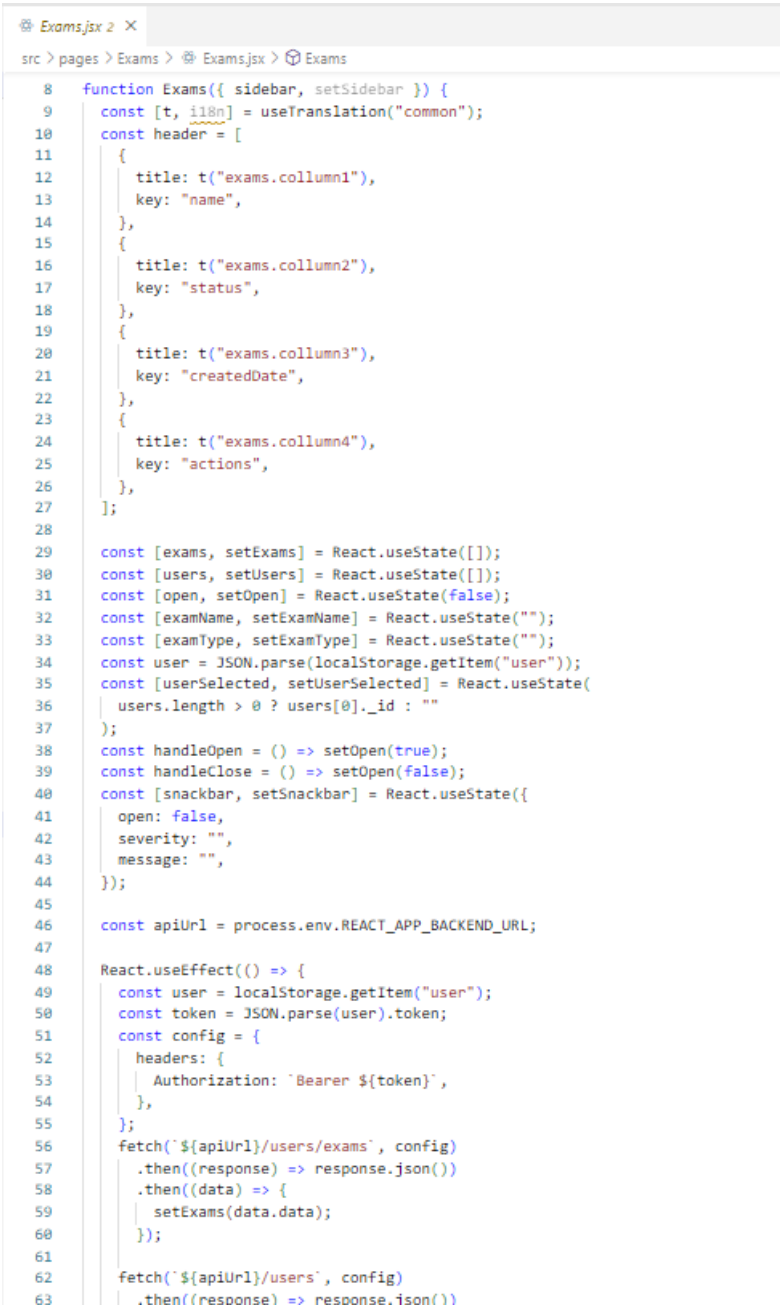


Figura 6.30 - Código da página Exames (Parte 1)

Na Figura 6.30 são criadas algumas constantes como vemos na linha 9 a 46 que serão importantes para a integração do *backend* com o *frontend*, feitas pelo *developer* Hélder Gonçalves. Também são definidas as funções *handleOpen* e *handleClose* para abrir e fechar uma caixa de diálogo para criar um exame, que depois acordado para testar o funcionamento dos exames.

Figura 6.31 - Código da página Exames (Parte 2)

Na Figura 6.31 na linha 124 se o utilizador for administrador ele vai renderizar um título e um botão para criar um exame que vemos da linha 125 a 131. Caso não seja administrador só renderiza o título.

Na linha 136 é renderizado a tabela que foi criada anteriormente como foi visto na Figura 6.28 e na Figura 6.29.

Na linha 138 a 149 é renderizado um componente *ModalCreateExam* que exibe um formulário para criação de um novo exame. O componente *ModalCreateExam* tem propriedades para tratar de fechar o formulário e o estado aberto do formulário, a lista de utilizadores, uma função para tratar alterações nos campos do formulário, o utilizador selecionado, uma função para tratar a criação de um novo exame, o nome do exame, uma função para definir o nome do exame e uma função para definir o tipo de exame.

Por fim, é renderizado um *SnackbarComponent* na linha 150 a 156 que exibe uma notificação. O *SnackbarComponent* informa os utilizadores sobre um processo que um aplicativo executou ou irá executar. Este tem propriedades para a gravidade, a mensagem da notificação, uma função para tratar de fechar a notificação e o estado aberto da notificação.

Infelizmente como o projeto não é a versão completa, a versão feita para ecrãs de pequenas dimensões ainda não foi desenvolvida, assim o resultado apresentado para esses ecrãs é o mesmo que o para ecrãs de grandes dimensões como é visto na Figura 6.32.

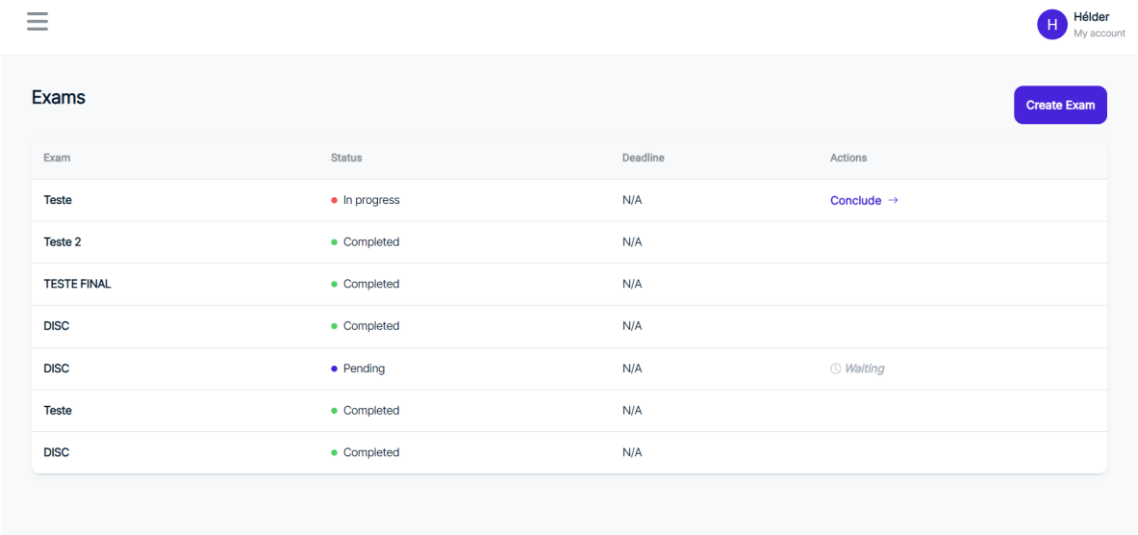


Figura 6.32 - Página Exames

6.5 Implementação da página Utilizadores

A quarta página a ser implementada foi a página Utilizadores. O estagiário para esta página teve a indicação que não teria o *layout* da página na aplicação *Figma*, logo a indicação que lhe é dada foi para fazer uma página do mesmo estilo que a exames com o elemento da tabela e um botão com a opção de criar um utilizador, como se vê na Figura 6.33. Esta página só é possível ser acedida por administradores, isto é, *developers* e membros da BitSapiens.

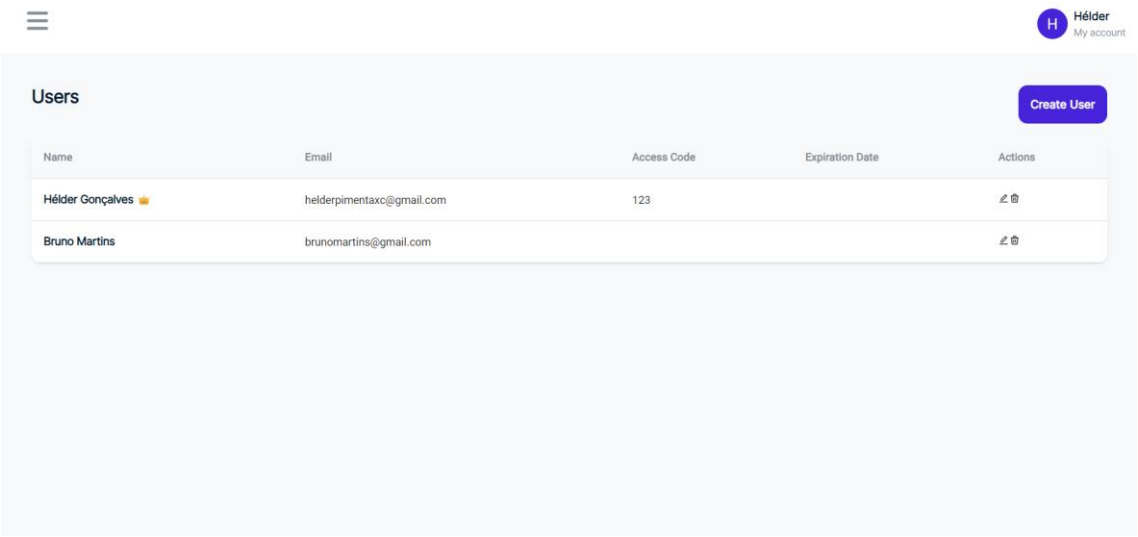


Figura 6.33 - Página Utilizadores

Na Figura 6.33 é possível observar que as únicas diferenças entre esta página e a de Exames que vemos na Figura 6.32 é os dados apresentados na tabela e que o botão cria um utilizador ao invés de um exame.

6.6 Implementação da página de exame

A quinta e sexta página a ser implementadas foram as páginas onde o questionário irá ser respondido pelo utilizador. O estagiário através da aplicação *Figma* vai usar a *template* que se pretende para aquela página para ecrãs de grandes ou pequenas dimensões como se visualiza na Figura 6.34.

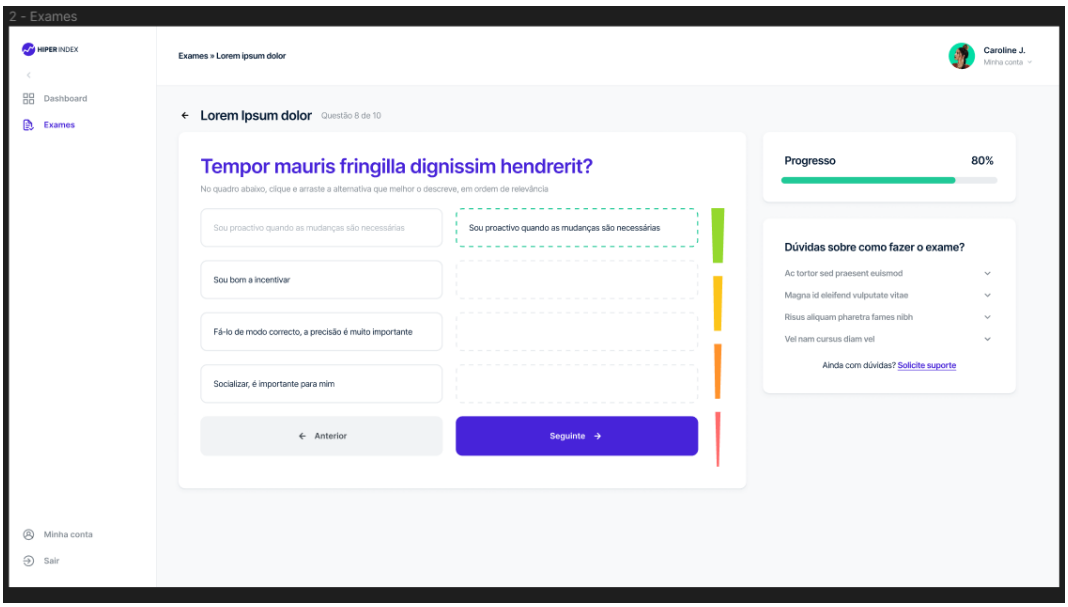


Figura 6.34 – Primeiro tipo de página de exame no Figma para grandes ou pequenas dimensões

A página na Figura 6.34 é onde existe um elemento muito importante chamado “drag and drop” que podemos visualizar dentro da pasta “components\Cards\ExamQuestion\DragAndDrop” o código seguinte como vemos na Figura 6.35.

index.jsx 1 ×

src > components > Cards > ExamQuestion > DragAndDrop > index.jsx > DragAndDrop

```
13  function DragAndDrop({
14    question,
15    saveAnswer,
16    backAnswer,
17    questionNumber,
18    questionLength,
19    finishExam,
20    leftItems,
21    rightItems,
22    setLeftItems,
23    setRightItems,
24  }) {
25    const [t, i18n] = useTranslation("common");
26    const handleDragEnd = (result) => {
27      if (!result.destination) return;
28
29      if (
30        result.source.droppableId === "left" &&
31        result.destination.droppableId.startsWith("right")
32      ) {
33        const newItem = { ...leftItems[result.source.index] };
34        const targetIndex = result.destination.index;
35        const rightCopy = [...rightItems];
36
37        // If there's already an item at the target position in the right column,
38        // move that item back to the left column
39        if (rightCopy[targetIndex].name) {
40          setLeftItems((prevLeft) => [...prevLeft, rightCopy[targetIndex]]);
41        }
42
43        // Set the new item in the right column
44        rightCopy[targetIndex] = newItem;
45        setRightItems(rightCopy);
46
47        // Remove the item from the left column
48        setLeftItems((prevLeft) =>
49          prevLeft.filter((_, index) => index !== result.source.index)
50        );
51      }
52
53      if (
54        result.source.droppableId.startsWith("right") &&
55        result.destination.droppableId.startsWith("right")
56      ) {
57        const temp = [...rightItems];
58        const [reorderedItem] = temp.splice(result.source.index, 1);
59        temp.splice(result.destination.index, 0, reorderedItem);
60        setRightItems(temp);
61      }
62    };
```

Figura 6.35 - Código do elemento Drag and Drop (Parte 1)

Na Figura 6.35 a função de *DragAndDrop* tem as seguintes propriedades: *question, saveAnswer, backAnswer, questionNumber, questionLength, finishExam, leftObjetos, rightObjetos, setLeftObjetos e setRightObjetos*.

Na linha 26 a 62 é criada uma variável chamada *handleDragEnd* que é chamada quando uma operação de arrastar e soltar termina. Também é verificado se a operação de arrastar

terminou em um destino válido. Caso contrário, ele retorna imediatamente ao local inicial. Se foi arrastado da coluna da esquerda para a coluna da direita, ele cria uma cópia do que foi arrastado, verifica se já existe o que foi arrastado na posição de destino na coluna da direita e, em caso afirmativo, move se lo de volta para a coluna da esquerda. Em seguida, é definido o novo objeto na coluna da direita e remove o objeto da coluna da esquerda. Se um objeto foi arrastado na coluna direita, ele reordena os itens na coluna direita com base na origem e no destino da operação de arrastar.

Na Figura 6.36, Figura 6.37 e Figura 6.38 é mostrado o código que irá mostrar o elemento de "*Drag and Drop*" através da linguagem HTML e CSS.

Figura 6.36 - Código do elemento Drag and Drop (Parte 2)

index.jsx 1

src > components > Cards > ExamQuestion > DragAndDrop > index.jsx > DragAndDrop

```
111                 <div className={styles.rightColumn}>
112                   {rightItems.map((item, index) => (
113                     <Droppable
114                       key={item.id}
115                       droppableId={`right-${item.id}`}
116                       direction="vertical"
117                     >
118                       {(provided) => (
119                         <div
120                           {...provided.droppableProps}
121                           ref={provided.innerRef}
122                           className={styles.rightDragOptionContainer}
123                         >
124                           <Draggable
125                             key={item.id}
126                             draggableId={item.id.toString()}
127                             index={index}
128                           >
129                             {(provided) => (
130                               <div
131                                 {...provided.draggableProps}
132                                 {...provided.dragHandleProps}
133                                 ref={provided.innerRef}
134                               >
135                                 {item.name && <OptionCard text={item.name} />}
136                               </div>
137                             )}
138                           </Draggable>
139                         </div>
140                       )}
141                     </Droppable>
142                   ))}
143                 </div>
144               </DragDropContext>
145             </div>
```

Figura 6.37 - Código do elemento Drag and Drop (Parte 3)

Figura 6.38 - Código do elemento Drag and Drop (Parte 4)

Na linha 76 é configurado um *DragDropContext* que envolve duas colunas como vimos na Figura 6.34.

Na linha 80 a 110 da Figura 6.36 temos a coluna da esquerda que contém uma lista de respostas que são arrastáveis. Cada objeto é renderizado como um componente *OptionCard* com o nome do objeto como texto. Na linha 111 a 145 da Figura 6.37 tem a coluna da direita que contém uma lista de áreas onde as respostas são largadas. Cada objeto é renderizado como um componente *OptionCard* com o nome do objeto como texto.

Depois na Figura 6.38 na linha 146 a 177 é renderizado uma linha de botões que o objeto ao utilizador navegar até a questão anterior, salvar sua resposta e navegar para a próxima questão ou finalizar o exame.

Por fim na linha 178 a 184 da Figura 6.38 é renderizado uma coluna de imagens para indicar ao utilizador a ordenação de respostas.

Tendo o elemento concluído, este é importado então para o ficheiro onde contém o código restante na pasta "*pages\Question\QuestionDragandDrop*" como vemos na Figura 6.39.

Question.jsx 3 ×

src > pages > Question > QuestionDragandDrop > Question.jsx > Question

```
271  return (
272    <>
273      <div
274        className={
275          sidebar ? styles.questionContainer : styles.questionContainerActive
276        }
277      >
278        <div className={styles.headerRow}>
279          <FaIcons.FaArrowLeft
280            className={styles.backIcon}
281            onClick={() => window.history.back()}
282          />
283          <h4>{exam.name}</h4>
284          <span className={styles.questionNumber}>
285            {t("question_drag_drop.question")} {questionNumber}{" "}
286            {t("question_drag_drop.of")} {questionLength}
287          </span>
288        </div>
289        <div className={styles.questionMainContainer}>
290          <DragAndDrop
291            question={exam.exam[questionNumber - 1]} // Assuming questionNumber is 1-based
292            saveAnswer={saveAnswer}
293            backAnswer={backAnswer}
294            questionNumber={questionNumber}
295            questionLength={questionLength}
296            finishExam={finishExam}
297            leftItems={leftItems}
298            rightItems={rightItems}
299            setLeftItems={setLeftItems}
300            setRightItems={setRightItems}
301          />
302          <div className={styles.questionContainer3}>
303            <ExamProgress progress={progress} />
304            <ExamDoubts />
305          </div>
306        </div>
307      </div>
```

Figura 6.39 - Código da página de exame com elemento Drag and Drop

Do código da página de exame de *Drag and Drop* é apenas mostrado alguma parte do código, pois o código restante foi desenvolvido pelo *developer* Hélder Gonçalves. Assim na Figura 6.39 está apresentado apenas o código que o estagiário desenvolveu para a página de exame de *Drag and Drop*.

Da linha 278 a 288 é renderizado uma linha de cabeçalho com um ícone de seta para trás, o nome do exame e o número da pergunta atual do número total de perguntas. Depois na linha 289 a 301 é renderizado o componente *DragAndDrop*.

Por fim na linha 302 a 305 é renderizado um componente *ExamProgress*, que exibe o progresso do utilizador no exame, e um componente *ExamDoubts*, que permite ao utilizador visualizar dúvidas frequentes.

Assim com isto, é apresentado o seguinte resultado visto na Figura 6.40 para ecrãs de grandes dimensões.

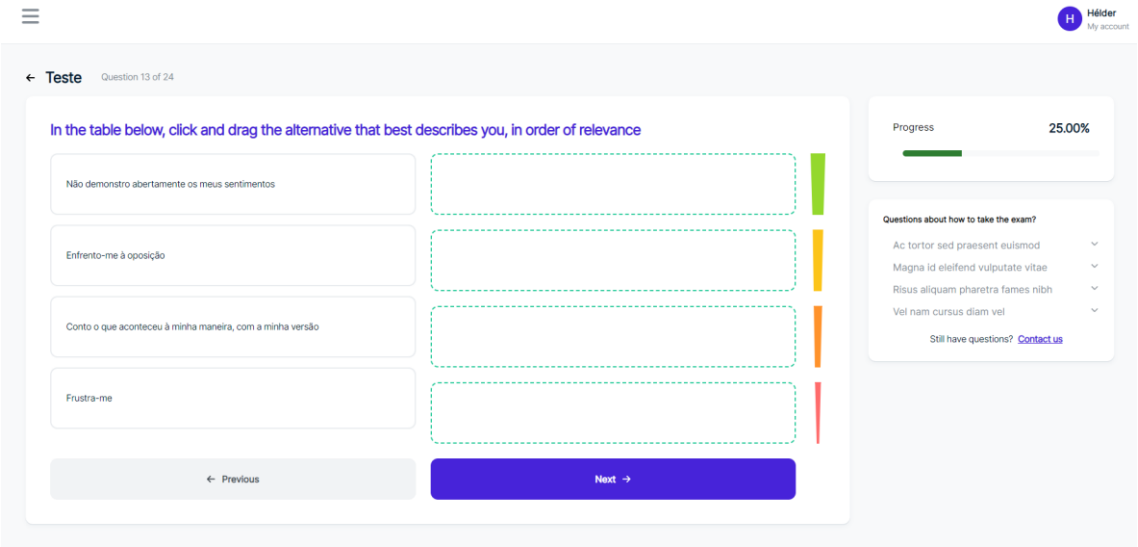


Figura 6.40 - Página de exame Drag and Drop

6.7 Implementação da página Meu Perfil

A sétima e última página a ser implementada foi a página Meu perfil. O estagiário através da aplicação *Figma* vai usar a *template* que se pretende para aquela página como se vê na Figura 6.41.

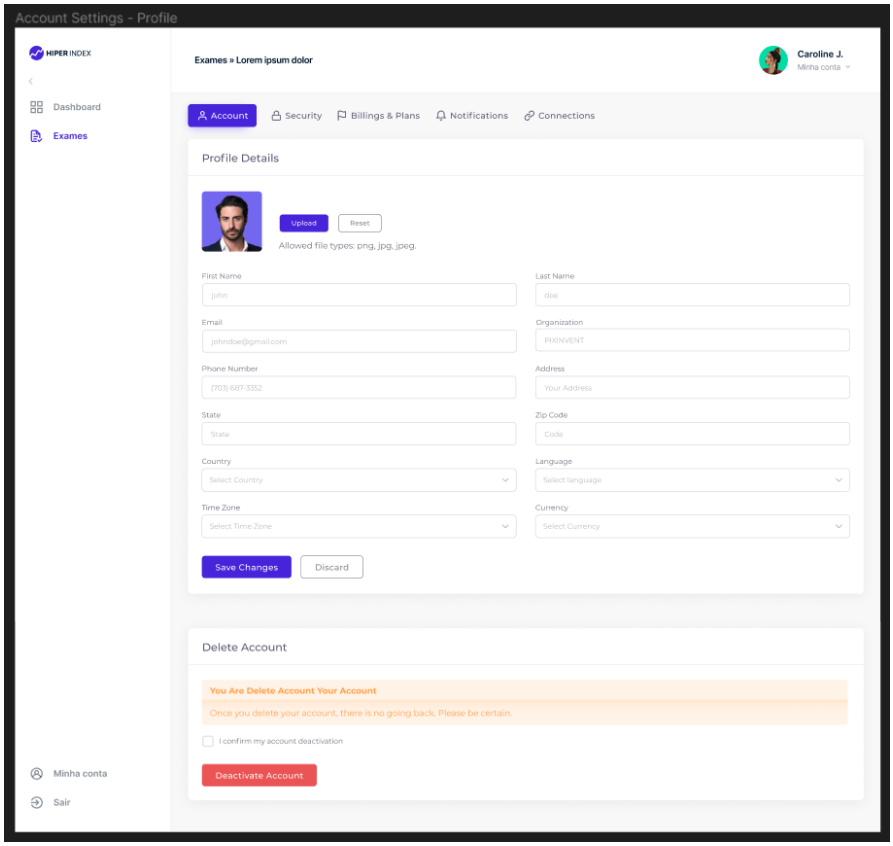


Figura 6.41 - Página Meu Perfil no Figma

O elemento novo apresentado na Figura 6.41 é a *TabBar*. A *TabBar* organiza e permite a navegação entre grupos de conteúdo relacionados e no mesmo nível de hierarquia.

Foi então desenvolvida na pasta "*components\AppBar\UserProfile*" através do seguinte código apresentado na Figura 6.42.

index.jsx 4

src > components > AppBar > UserProfile > index.jsx > ...

```
13  const TabsUserProfile = ({ value, handleChange }) => {
14    const [t, i18n] = useTranslation("common");
15    return (
16      <Box sx={{ width: "100%", bgcolor: "transparent", display: "flex" }}>
17        <Tabs
18          value={value}
19          onChange={handleChange}
20          variant="scrollable"
21          scrollButtons
22          allowScrollButtonsMobile
23          aria-label="icon position tabs example"
24          sx={{ bgcolor: "transparent" }}
25        >
26          <Tab
27            icon={<AiIcons.AiOutlineUser />}
28            iconPosition="start"
29            label={t("profile.tab_collumn1")}
30          />
31          <Tab
32            icon={<AiIcons.AiOutlineLock />}
33            label={t("profile.tab_collumn2")}
34            iconPosition="start"
35          />
36          <Tab
37            icon={<BsIcons.BsCreditCard />}
38            iconPosition="start"
39            label={t("profile.tab_collumn3")}
40          />
41          <Tab
42            icon={<BsIcons.BsBell />}
43            label={t("profile.tab_collumn4")}
44            iconPosition="start"
45          />
46          <Tab
47            icon={<FiIcons.FiLink />}
48            iconPosition="start"
49            label={t("profile.tab_collumn5")}
50          />
51        </Tabs>
52      </Box>
53    );
54  };
55
56  export default TabsUserProfile;
```

Figura 6.42 - Código do elemento TabBar

Na Figura 6.42 na linha 16 é declarado um *Box*, que é essencialmente um div com alguns acessórios adicionais para estilo. A *Box* está configurada para ocupar 100% da largura de seu pai, ter um fundo transparente e exibir seus filhos em uma linha usando um *flexbox*.

Na linha 17 é declarado o componente *Tabs* onde vai ser customizado pelas propriedades deste componente.

Por fim é renderizado cinco componentes *Tab* dentro do componente *Tabs* da linha 26 a 50. Cada um destes têm um nome e um ícone que pode ser customizado com a propriedade *iconPosition*.

Na Figura 6.43 é visualizado o código onde vai ser implementado a *TabBar* na pasta "*pages\Profile*".

Profile.jsx 2

src > pages > Profile > Profile.jsx > Profile

```
53  function Profile({ sidebar, setSidebar }) {
54    const [t, i18n] = useTranslation("common");
55    const user = JSON.parse(localStorage.getItem("user") || "{}");
56    const [updatedProfile, setUpdatedProfile] = useState({});
57    const [open, setOpen] = useState(true);
58
59    useEffect(() => {
60      setOpen(false);
61      console.log(user);
62    }, []);
63
64    const handleInputChange = (e, field) => {
65      setUpdatedProfile({
66        ...updatedProfile,
67        [field]: e.target.value,
68      });
69    };
70
71    const theme = useTheme();
72    const [value, setValue] = useState(0);
73
74    const handleChange = (event, newValue) => {
75      console.log(newValue);
76      setValue(newValue);
77    };
78
79    if (open) {
80      return (
81        <Backdrop
82          sx={{ color: "#fff", zIndex: (theme) => theme.zIndex.drawer + 1 }}
83          open={open}
84        >
85          <CircularProgress color="inherit" />
86        </Backdrop>
87      );
88    }
89
90    const openFilePicker = () => {
91      document.getElementById("file").click();
92    };
93
```

Figura 6.43 - Código para página Meu Perfil (Parte 1)

Na Figura 6.43 serão apenas estabelecidas algumas das variáveis e funções do código, como na linha 55 é recuperado os dados do utilizador do armazenamento local. É definido na linha 49 a 62 o *useEffect* que é executado uma vez quando o componente é montado para registar os dados do utilizador. Na linha 64 a 69 o *handleInputChange* atualiza o *updateProfile* quando o utilizador altera o valor de um campo de entrada.

Na Figura 6.44, Figura 6.45, Figura 6.46 e Figura 6.47 é mostrado o código que irá mostrar a página Meu Perfil e a implementação da *TabBar* através da linguagem HTML e CSS.

Profile.jsx 2

src > pages > Profile > Profile.jsx > Profile

```jsx
return (
  <div
    className={
      sidebar ? styles.profileContainer : styles.profileContainerActive
    }
  >
    <Container>
      <TabsUserProfile value={value} handleChange={handleChange} />
      <TabPanel
        value={value}
        index={0}
        dir={theme.direction}
        style={{ width: "100%" }}
      >
        <div className={styles.profileCard}>
          <div className={styles.profileHeader}>
            <h4>{t("profile.title")}</h4>
          </div>
          <div className={styles.avatarEdit}>
            <div className={styles.avatarcolumn1}>
              <Avatar
                sizes="large"
                sx={{
                  width: 100,
                  height: 100,
                  marginLeft: "24px",
                }}
              >
                {user && user.firstName[0] + user.lastName[0]}
              </Avatar>
            </div>
            <div className={styles.avatarcolumn2}>
              <input
                type="file"
                id="file"
                style={{ display: "none" }}
                accept="image/*"
                onChange={(e) => console.log(e.target.files[0])}
              />
              <button className={styles.upload} onClick={openFilePicker}>
                {t("profile.button_upload")}
              </button>
              <button className={styles.reset}>
                {t("profile.button_reset")}
              </button>
              <p>{t("profile.avatar_description")}</p>
            </div>
          </div>
```

Figura 6.44 - Código para página Meu Perfil (Parte 2)

Profile.jsx 2

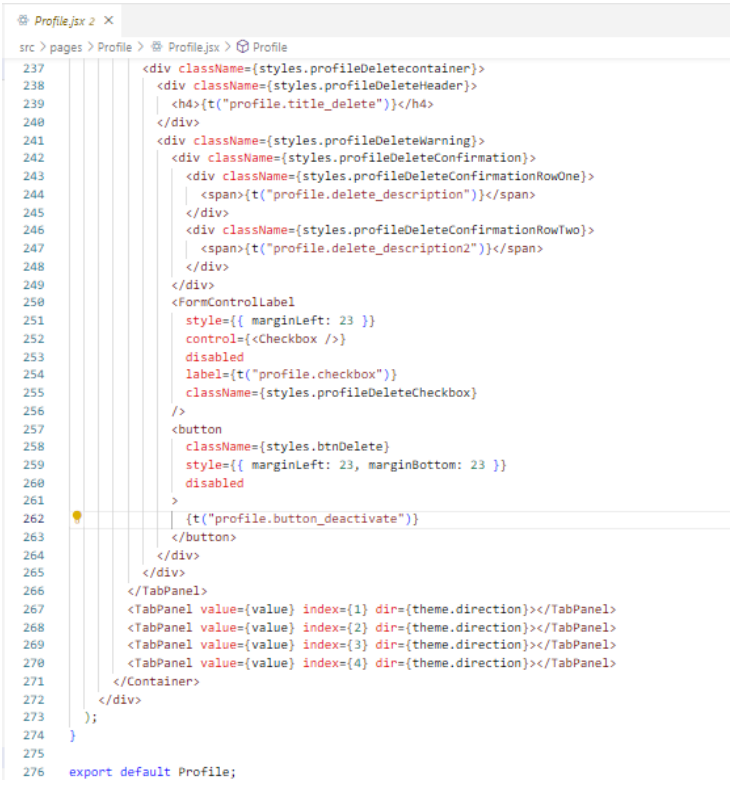
src > pages > Profile > Profile.jsx > Profile

```jsx
          <div className={styles.profileForm}>
            <div className={styles.profileFormLine}>
              <input
                type="text"
                placeholder={t("profile.first_name")}
                value={{ user } && user.firstName}
                onChange={(e) => handleInputChange(e, "firstName")}
              />
              <input
                type="text"
                placeholder={t("profile.last_name")}
                value={{ user } && user.lastName}
                onChange={(e) => handleInputChange(e, "lastName")}
              />
            </div>
            <div className={styles.profileFormLine}>
              <input
                type="email"
                placeholder={t("profile.email")}
                value={{ user } && user.email}
                disabled
              />
              <input
                type="text"
                placeholder={t("profile.organization")}
                value={user && user.organization}
                onChange={(e) =>
                  setUpdatedProfile({
                    ...updatedProfile,
                    organization: e.target.value,
                  })
                }
              />
            </div>
            <div className={styles.profileFormLine}>
              <input
                type="text"
                placeholder={t("profile.phone")}
                value={user && user.phoneNumber}
                onChange={(e) => handleInputChange(e, "phoneNumber")}
              />
              <input
                type="text"
                placeholder={t("profile.address")}
                value={user && user.address}
                onChange={(e) => handleInputChange(e, "address")}
              />
            </div>
```

Figura 6.45 - Código para página Meu Perfil (Parte 3)

```
Profile.jsx 2
src > pages > Profile > Profile.jsx > Profile
                <div className={styles.profileFormLine}>
                  <input
                    type="text"
                    placeholder={t("profile.state")}
                    value={(updatedProfile && updatedProfile.state) || ""}
                    onChange={(e) => handleInputChange(e, "state")}
                  />
                  <input
                    type="text"
                    placeholder={t("profile.zip_code")}
                    value={user && user.zipCode}
                    onChange={(e) => handleInputChange(e, "zipCode")}
                  />
                </div>
                <div className={styles.profileFormLine}>
                  <input
                    type="text"
                    placeholder={t("profile.country")}
                    value={user && user.country}
                    onChange={(e) => handleInputChange(e, "country")}
                  />
                  <input
                    type="text"
                    placeholder={t("profile.language")}
                    value={user && user.language}
                    onChange={(e) => handleInputChange(e, "language")}
                  />
                </div>
                <div className={styles.profileFormLine}>
                  <input
                    type="text"
                    placeholder={t("profile.time_zone")}
                    value={user && user.timeZone}
                    onChange={(e) => handleInputChange(e, "timeZone")}
                  />
                  <input type="text" placeholder={t("profile.currency")} />
                </div>
                <div className="">
                  <button className={styles.upload}>
                    {t("profile.button_save")}
                  </button>
                  <button className={styles.reset}>
                    {t("profile.button_discard")}
                  </button>
                </div>
              </div>
            </div>
```

Figura 6.46 - Código para página Meu Perfil (Parte 4)



```
Profile.jsx 2
src > pages > Profile > Profile.jsx > Profile
            <div className={styles.profileDeletecontainer}>
              <div className={styles.profileDeleteHeader}>
                <h4>{t("profile.title_delete")}</h4>
              </div>
              <div className={styles.profileDeleteWarning}>
                <div className={styles.profileDeleteConfirmation}>
                  <div className={styles.profileDeleteConfirmationRowOne}>
                    <span>{t("profile.delete_description")}</span>
                  </div>
                  <div className={styles.profileDeleteConfirmationRowTwo}>
                    <span>{t("profile.delete_description2")}</span>
                  </div>
                </div>
                <FormControlLabel
                  style={{ marginLeft: 23 }}
                  control={<Checkbox />}
                  disabled
                  label={t("profile.checkbox")}
                  className={styles.profileDeleteCheckbox}
                />
                <button
                  className={styles.btnDelete}
                  style={{ marginLeft: 23, marginBottom: 23 }}
                  disabled
                >
                  {t("profile.button_deactivate")}
                </button>
              </div>
            </div>
          </TabPanel>
          <TabPanel value={value} index={1} dir={theme.direction}></TabPanel>
          <TabPanel value={value} index={2} dir={theme.direction}></TabPanel>
          <TabPanel value={value} index={3} dir={theme.direction}></TabPanel>
          <TabPanel value={value} index={4} dir={theme.direction}></TabPanel>
        </Container>
      </div>
    );
  }

  export default Profile;
```

Figura 6.47 - Código para página Meu Perfil (Parte 5)

Na Figura 6.44 na linha 100 é renderizado um componente *Container*, que é essencialmente uma *div* com alguns acessórios adicionais para estilo, e que no caso irá ajudar com a *TabBar*.

O componente *TabsUserProfile* é então importado da pasta "*components\AppBar\UserProfile*" para a linha 101 como vemos na Figura 6.44

Da linha 108 até à linha 240 através da Figura 6.44, Figura 6.45, Figura 6.46 e Figura 6.47 é feito um título e um formulário para atualização do perfil do utilizador. O formulário contém campos de entrada para nome, sobrenome, e-mail, organização, número de telefone, endereço, estado, CEP, país, idioma e fuso horário do utilizador. O campo email está desabilitado e os outros campos possuem manipuladores *onChange* que atualizam o utilizador ou o estado *UpdateProfile* quando o utilizador altera o valor do campo. Também tem uma seção para *upload* de um novo avatar. O avatar atual do utilizador é exibido como um componente avatar, e o utilizador pode fazer *upload* de um novo avatar.

Por fim na linha 241 a 265 na Figura 6.47 tem uma seção para excluir a conta do utilizador. O utilizador deve marcar uma caixa de seleção e clicar em um botão para excluir sua conta, mas tanto a caixa de seleção quanto o botão estão desativados no momento.

Assim com isto, é apresentado o seguinte resultado visto na Figura 6.48 e Figura 6.49 para ecrãs de grandes dimensões ou pequenas dimensões.

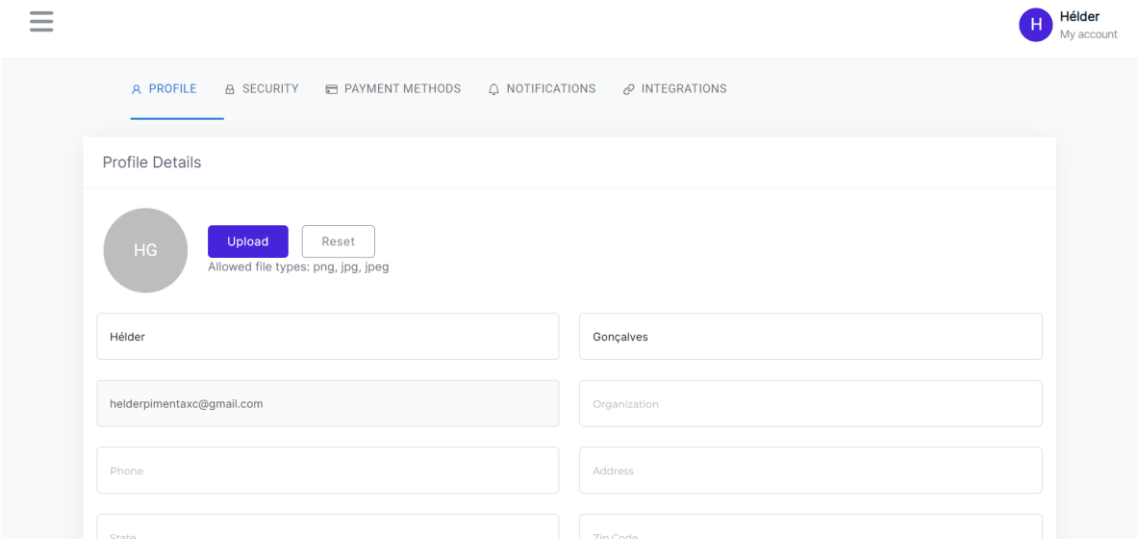


Figura 6.48 - Página de Meu Perfil (Parte 1)

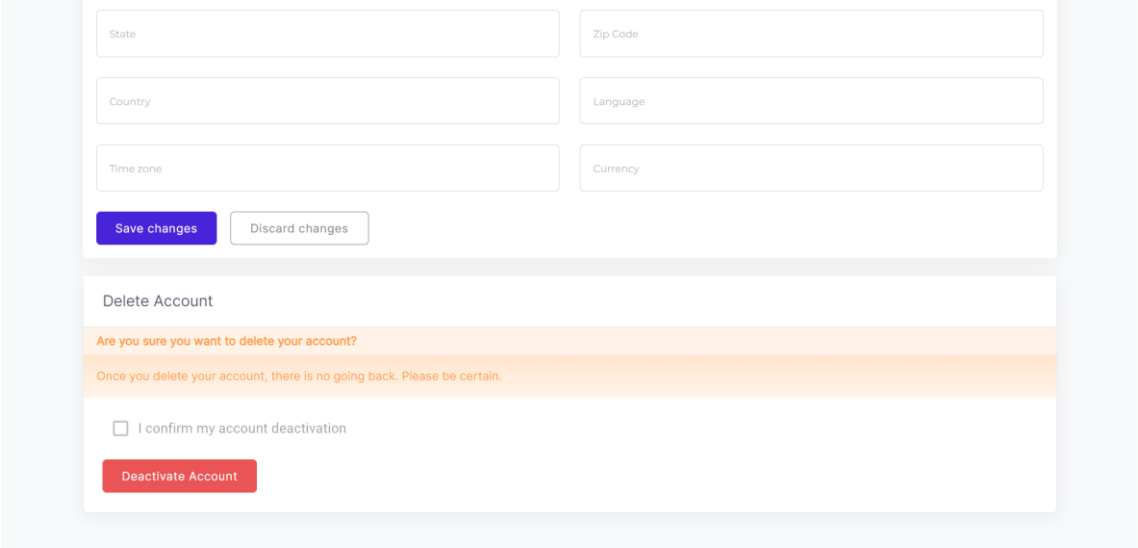


Figura 6.49 - Página de Meu Perfil (Parte 2)

7 Verificação e Validação

Ao longo do desenvolvimento da aplicação foram realizados testes de forma incremental, ou seja, à medida que as funcionalidades eram desenvolvidas eram realizados testes de modo a confirmar que estavam corretas, ou que melhorias podiam ser desenvolvidas.

Como a aplicação está longe de estar terminada completamente, apenas só foram testes unitários aos elementos que o estagiário e o *developer* Hélder Gonçalves iriam adicionando à aplicação. Por exemplo se o estagiário a testar a *Sidebar*, experimentava se estava tudo operacional dentro de todas as possibilidades que existem para aquele elemento, como é visto na Tabela 7.1.

Tabela 7.1 - Casos de Teste - SideBar

ID	Entrada	Observações	Resultados Esperados	V	F
1	Selecionar o ícone na Navbar para abrir a Sidebar	O utilizador seleciona a opção do ícone na Navbar	Sidebar abre ou fecha-se corretamente para o utilizador	V	
2	Selecionar a opção Painel	O utilizador já estar na página Painel	A Sidebar fecha-se e o utilizador permanece na mesma página	V	
3	Selecionar a opção Painel	O utilizador estar noutra página da aplicação	A Sidebar fecha-se e o utilizador vai ter à página Painel	V	
4	Selecionar nenhuma opção na Sidebar	O utilizador tem a Sidebar aberta	A Sidebar fecha-se e o utilizador permanece na mesma página	V	

8 Conclusão

Este projeto teve como objetivo o desenvolvimento de uma aplicação web, para ajudar empresas e organizações a ter uma melhor produtividade e eficiência através dos exames DISC que irão fornecer análises completas e detalhadas dos seus trabalhadores ajudando a produtividade da empresa e até mesmo no recrutamento de novos funcionários.

Numa primeira fase, em simultâneo com a integração do estagiário com a empresa, realizou-se a analise de especificações do projeto onde o orientador do projeto já tinha alguns projetos em mente para o estagiário. Durante o desenvolvimento teve-se em atenção desenvolver aplicação para vários tipos de ecrãs de dispositivos eletrónicos e onde também foram adicionadas novas funcionalidades tendo em consideração os vários cenários que podem ocorrer enquanto o utilizador utiliza a aplicação.

No que diz respeito aos testes, estes foram realizados de forma incremental à medida que aplicação era desenvolvida em conjunto com o *developer* Hélder Gonçalves.

A aplicação encontra-se ainda em desenvolvimento de momento como foi referido neste relatório só e pode ser utilizada pelos utilizadores da BitSapiens. O orientador do projeto quer continuar tem intenções de continuar o projeto com o estagiário e o *developer* Hélder Gonçalves.

Bibliografia

Analise de requisitos - InfoEscola. (s.d.). Obtido de InfoEscola: https://www.infoescola.com/engenharia-de-software/analise-de-requisitos/

Bitsapiens. (2021). *Bitsapiens - About*. Obtido de Bitsapiens: https://bitsapiens.io/bitsapiens-about-us/

Gackenheimer, C. (2 de Setembro de 2015). Introduction to React.

Jasmin Software - Metodologia Agile. (2023). Obtido de Jasmin Software: https://www.jasminsoftware.pt/blog/metodologia-agile/

Luz.vc - ClickUp. (s.d.). Obtido de ClickUp: https://luz.vc/clickup

Nakagawa, P. D. (21 de Agosto de 2017). *Casos de Uso e Diagrama de Casos de Uso.* Obtido de https://edisciplinas.usp.br/pluginfile.php/3720765/course/section/857581/Aula02_CasosDeUso.pdf

Sólides - DISC. (2023). Obtido de Sólides: https://blog.solides.com.br/analisar-perfil-disc/

The $2 Billion Question of Who You Are at Work. (5 de Março de 2023). Obtido de The New York Times: https://www.nytimes.com/2023/03/05/business/remote-work-personality-tests.html

Wikipédia - Cascading Style Sheets. (s.d.). Obtido de Wikipédia: https://pt.wikipedia.org/wiki/Cascading_Style_Sheets

Wikipédia - Figma. (s.d.). Obtido de Wikipédia: https://pt.wikipedia.org/wiki/Figma

Wikipédia - HTML. (s.d.). Obtido de Wikipédia: https://pt.wikipedia.org/wiki/HTML

Wikipédia - JavaScript. (s.d.). Obtido de Wikipédia: https://pt.wikipedia.org/wiki/JavaScript

Wikipédia - Node.js. (s.d.). Obtido de Wikipédia: https://pt.wikipedia.org/wiki/Node.js

Wikipédia - React. (s.d.). Obtido de Wikipédia: https://pt.wikipedia.org/wiki/React_(JavaScript)

Wikipédia - Visual Studio Code. (s.d.). Obtido de Wikipédia : https://pt.wikipedia.org/wiki/Visual_Studio_Code

Anexos

Anexos 1: Implementação

Anexos 1.1: Código para App.js

```
import React, { useState } from "react";
import {
  Routes,
  Route,
  Navigate,
  useNavigate,
  useLocation,
} from "react-router-dom";
import Login from "./pages/Login/Login";
import Home from "./pages/Home/Home";
import Navbar from "./components/Navbar";
import Profile from "./pages/Profile/Profile";
import Exams from "./pages/Exams/Exams";
import Question from "./pages/Question/QuestionDragandDrop/Question";
import Question2 from "./pages/Question/QuestionPointandClick/Question";
import Users from "./pages/Users/Users";
import "./App.css";
import Error from "./pages/Error/Error";
import { history } from "./_helpers";
import { PrivateRoute } from "./components/PrivateRoute";

function App() {
  // init custom history object to allow navigation from
  // anywhere in the react app (inside or outside components)
  history.navigate = useNavigate();
  history.location = useLocation();
  const [sidebar, setSidebar] = useState(false);

  return (
    <>
      <Navbar sidebar={sidebar} setSidebar={setSidebar} />
      <Routes>
        <Route
          path="/"
          element={
            <PrivateRoute>
              <Home sidebar={sidebar} setSidebar={setSidebar} />
            </PrivateRoute>
          }
        />
```

```
<Route
  path="/exam/:examId"
  element={
    <PrivateRoute>
      <Question sidebar={sidebar} setSidebar={setSidebar} />
    </PrivateRoute>
  }
/>
<Route
  path="/question2"
  element={
    <PrivateRoute>
      <Question2 sidebar={sidebar} setSidebar={setSidebar} />
    </PrivateRoute>
  }
/>
<Route
  path="/exams"
  element={
    <PrivateRoute>
      <Exams sidebar={sidebar} setSidebar={setSidebar} />
    </PrivateRoute>
  }
/>
<Route
  path="/profile"
  element={
    <PrivateRoute>
      <Profile sidebar={sidebar} setSidebar={setSidebar} />
    </PrivateRoute>
  }
/>
<Route
  path="/users"
  element={
    <PrivateRoute>
      <Users sidebar={sidebar} setSidebar={setSidebar} />
    </PrivateRoute>
  }
/>
<Route
  path="*"
  element={
    <PrivateRoute>
      <Error sidebar={sidebar} setSidebar={setSidebar} />
    </PrivateRoute>
  }
/>
<Route path="/login" element={<Login />} />
```

```
        <Route path="/" element={<Navigate replace to="/login" />} />
      </Routes>
    </>
  );
}

export { App };
```

Anexos 1.2: Código para index.js

```
import React from "react";
import { createRoot } from "react-dom/client";
import { Provider } from "react-redux";
import { BrowserRouter } from "react-router-dom";
import { store } from "./_store";
import { App } from "./App";
import "./index.css";
import { I18nextProvider } from "react-i18next";
import i18next from "i18next";
import common_pt from "./translations/pt/common.json";
import common_en from "./translations/en/common.json";

i18next.init({
  interpolation: { escapeValue: false }, // React already does escaping
  lng: "en", // language to use
  resources: {
    en: {
      common: common_en, // 'common' is our custom namespace
    },
    pt: {
      common: common_pt,
    },
  },
});

const container = document.getElementById("root");
const root = createRoot(container);

root.render(
  <I18nextProvider i18n={i18next}>
    <Provider store={store}>
      <BrowserRouter>
        <App />
      </BrowserRouter>
    </Provider>
  </I18nextProvider>
);
```

Anexos 1.3: Código de translação para inglês

```
{
  "login": {
    "title": "Login",
    "email": "Email",
    "password": "Password",
    "login": "Login",
    "forgot_password": "Forgot Password?",
    "remember_me": "Remember Me",
    "i_have_a_code": "I have a code",
    "sign_up": "Sign Up",
    "sign_up_link": "Don't have an account?",
    "insert_code": "Insert your code",
    "access_with_code": "Access with code",
    "already_have_an_account": "Already have an account?",
    "sign_in": "Sign In",
    "create_account": "Create Account",
    "name": "Name",
    "confirm_password": "Confirm Password",
    "dont_have_an_account": "Don't have an account?"
  },
  "dashboard": {
    "title": "Dashboard"
  },
  "home": {
    "title": "My exams",
    "button_see": "See all",
    "almost_there": "You're almost there!",
    "almost_there_description": "Not long to complete your exams",
    "button_finish": "Complete Exams",
    "progress": "Progress"
  },
  "exams": {
    "title": " My Exams",
    "admin_title": "Exams",
    "collumn1": "Exam",
    "collumn2": "Status",
    "collumn3": "Deadline",
    "collumn4": "Actions",
    "button_create": "Create Exam",
    "waiting": "Waiting",
    "conclude": "Conclude",
    "completed": "Completed",
    "in_progress": "In progress",
    "pending": "Pending"
```

```
  },
  "profile": {
    "tab_collumn1": "Profile",
    "tab_collumn2": "Security",
    "tab_collumn3": "Payment Methods",
    "tab_collumn4": "Notifications",
    "tab_collumn5": "Integrations",
    "title": "Profile Details",
    "button_upload": "Upload",
    "button_reset": "Reset",
    "avatar_description": "Allowed file types: png, jpg, jpeg",
    "first_name": "First Name",
    "last_name": "Last Name",
    "email": "Email",
    "organization": "Organization",
    "phone": "Phone",
    "address": "Address",
    "state": "State",
    "zip_code": "Zip Code",
    "country": "Country",
    "language": "Language",
    "time_zone": "Time zone",
    "currency": "Currency",
    "button_save": "Save changes",
    "button_discard": "Discard changes",
    "title_delete": "Delete Account",
    "delete_description": "Are you sure you want to delete your
account?",
    "delete_description2": "Once you delete your account, there is no
going back. Please be certain.",
    "checkbox": "I confirm my account deactivation",
    "button_deactivate": "Deactivate Account"
  },
  "users": {
    "title": "Users",
    "collumn1": "Name",
    "collumn2": "Email",
    "collumn3": "Access Code",
    "collumn4": "Expiration Date",
    "collumn5": "Actions",
    "button_create": "Create User"
  },
  "navbar": {
    "account": " My account"
  },
  "sidebar": {
    "dashboard": "Dashboard",
    "exams": "Exams",
    "users": "Users",
```

```
    "profile": " My Profile",
    "logout": "Logout"
  },
  "create_exam": {
    "title": "Create Exam",
    "name": "Exam name",
    "user": "User",
    "type": "Type of Exam",
    "button_create": "Create",
    "button_cancel": "Cancel"
  },
  "create_user": {
    "title": "Create User",
    "first_name": "First name",
    "last_name": "Last name",
    "email": "Email",
    "access_code": "Access code",
    "expiration_date": "Expiration date",
    "type": "Type of User",
    "administrator": "Administrator",
    "user": "User",
    "button_create": "Create",
    "button_cancel": "Cancel",
    "password": "Password"
  },
  "question_drag_drop": {
    "question": "Question",
    "of": "of",
    "instructions": "In the table below, click and drag the alternative
that best describes you, in order of relevance",
    "button_next": "Next",
    "button_previous": "Previous",
    "button_finish": "Finish",
    "progress": "Progress",
    "doubts_title": "Questions about how to take the exam?",
    "still_doubts": "Still have questions?",
    "doubts_contact": "Contact us"
  },
  "exam_finished": {
    "title": "The exam was completed successfully!",
    "title_dont_exist": "The exam does not exist!",
    "button_back": "Back to exams"
  }

}
```

Anexos 1.4: Código de translação para português


```
{
  "login": {
    "title": "Login",
    "email": "Email",
    "password": "Password",
    "login": "Login",
    "forgot_password": "Esqueceu sua senha?",
    "remember_me": "Lembrar-me",
    "i_have_a_code": "Já tenho um código",
    "sign_up": "Registar",
    "sign_up_link": "Ainda não tem uma conta?",
    "insert_code": "Digite o código",
    "access_with_code": "Acede com o código",
    "already_have_an_account": "Já tem uma conta?",
    "sign_in": "Entrar",
    "create_account": "Criar conta",
    "name": "Nome",
    "confirm_password": "Confirmar password",
    "dont_have_an_account": "Não tem uma conta?"
  },
  "dashboard": {
    "title": "Dashboard"
  },
  "home": {
    "title": "Meus exames",
    "button_see": "Ver todos",
    "almost_there": "Você está quase lá!",
    "almost_there_description": "Falta pouco para concluir seus exames",
    "button_finish": "Concluir Exames",
    "progress": "Progresso"
  },
  "exams": {
    "title": " Meus Exames",
    "admin_title": "Exames",
    "collumn1": "Exames",
    "collumn2": "Estado",
    "collumn3": "Prazo para conclusão",
    "collumn4": "Ações",
    "button_create": "Criar Exame",
    "waiting": "Aguardando",
    "conclude": "Concluir",
    "completed": "Concluído",
    "in_progress": "Em progresso",
    "pending": "Pendente"
  },
  "profile": {
```

```
    "tab_collumn1": "Perfil",
    "tab_collumn2": "Segurança",
    "tab_collumn3": "Métodos de Pagamento",
    "tab_collumn4": "Notificações",
    "tab_collumn5": "Integrações",
    "title": "Detalhes do Perfil",
    "button_upload": "Carregar",
    "button_reset": "Reiniciar",
    "avatar_description": "Tipos de arquivos permitidos: png, jpg, jpeg",
    "first_name": "Primeiro Nome",
    "last_name": "Ultimo Nome",
    "email": "Email",
    "organization": "Organização",
    "phone": "Telemóvel",
    "address": "Morada",
    "state": "Cidade",
    "zip_code": "Código Postal",
    "country": "País",
    "language": "Linguagem",
    "time_zone": "Fuso Horário",
    "currency": "Moeda",
    "button_save": "Guardar Alterações",
    "button_discard": "Descartar Alterações",
    "title_delete": "Desativar Conta",
    "delete_description": "Tem a certeza que deseja desativar a sua
conta?",
    "delete_description2": "Depois de excluir sua conta, não há como
voltar atrás. Por favor, tenha certeza.",
    "checkbox": "Confirmo a desativação da minha conta",
    "button_deactivate": "Desativar Conta"
  },
  "users": {
    "title": "Utilizador",
    "collumn1": "Nome",
    "collumn2": "Email",
    "collumn3": "Código de acesso",
    "collumn4": "Data de expiração",
    "collumn5": "Ações",
    "button_create": "Criar Utilizador"
  },
  "navbar": {
    "account": " Minha conta"
  },
  "sidebar": {
    "dashboard": "Painel",
    "exams": "Exames",
    "users": "Utilizadores",
    "profile": "Meu Perfil",
    "logout": "Sair"
```

```
  },
  "create_exam": {
    "title": "Criar Exame",
    "name": "Nome do exame",
    "user": "Utilizador",
    "type": "Tipo de exame",
    "button_create": "Criar",
    "button_cancel": "Cancelar"
  },
  "create_user": {
    "title": "Criar Utilizador",
    "first_name": "Primeiro Nome",
    "last_name": "Último Nome",
    "email": "Email",
    "access_code": "Codigo de acesso",
    "expiration_date": "Data de expiração",
    "type": "Tipo de utilizador",
    "administrator": "Administrador",
    "user": "Utilizador",
    "button_create": "Criar",
    "button_cancel": "Cancelar",
    "password": "Password"
  },
  "question_drag_drop": {
    "question": "Questão",
    "of": "de",
    "instructions": "No quadro abaixo, clique e arraste a alternativa que
melhor o descreve, em ordem de relevância",
    "button_next": "Seguinte",
    "button_previous": "Anterior",
    "button_finish": "Terminar",
    "progress": "Progresso",
    "doubts_title": "Dúvidas sobre como fazer o exame?",
    "still_doubts": "Ainda com dúvidas?",
    "doubts_contact": "Contacte-nos"
  },
  "exam_finished": {
    "title": "O exame foi concluído com sucesso!",
    "title_dont_exist": "O exame não existe!",
    "button_back": "Voltar para os exames"
  }
}
```

Anexos 1.5: Código de estilos CSS para a página Meu Perfil

```
@font-face {
  font-family: Inter;
```

```
  src: url(../../assets/fonts/Montserrat-Regular.ttf);
}

.profileContainer {
  display: flex;
  align-items: flex-start;
  align-content: flex-start;
  align-self: stretch;
  flex-wrap: wrap;
  margin-left: 250px;
  transition: 550ms;
}

.profileContainerActive {
  display: flex;
  align-items: flex-start;
  align-content: flex-start;
  align-self: stretch;
  flex-wrap: wrap;
  transition: 350ms;
}

.profileCard {
  display: flex;
  flex-direction: column;
  align-items: flex-start;
  align-content: flex-start;
  width: 100%;
  height: 100%;
  border-radius: 6px;
  background: rgba(0, 0, 0, 0);
  box-shadow: 0px 4px 24px 0px rgba(0, 0, 0, 0.06);
  background-color: white;
  margin: 0px 24px 0px 24px;
}

.profileHeader {
  display: flex;
  flex-direction: column;
  align-items: flex-start;
  justify-content: center;
  width: 100%;
  height: 60px;
  border-bottom: 1px solid var(--gray-gray-2, #e9ecef);
  background: white;
}

.profileHeader h4 {
  color: var(--1-theme-color-heading-display-text, #5e5873);
```

```
  font-feature-settings: "clig" off, "liga" off;
  font-size: 18px;
  font-style: normal;
  font-weight: 500;
  line-height: normal;
  margin-left: 23px;
}

.avatarEdit {
  display: flex;
  flex-direction: row;
  width: 100%;
  align-items: center;
  justify-content: flex-start;
  gap: 24px;
  margin-top: 24px;
  margin-bottom: 24px;
}

.avatarcolumn1 {
  display: flex;
  flex-direction: column;
  align-items: center;
}

.avatarcolumn2 {
  display: flexbox;
  align-items: center;
  justify-content: center;
}

.avatarcolum2 p {
  color: var(--1-theme-color-body-text, #6e6b7b);
  font-feature-settings: "clig" off, "liga" off;
  font-size: 14px;
  font-style: normal;
  font-weight: 400;
  line-height: 21px;
}

.upload {
  display: inline-flex;
  padding: 10px 22px;
  justify-content: center;
  align-items: center;
  gap: 8px;
  border-radius: 5px;
  background: var(--Accent, #4723d9);
  color: var(--1-theme-color-white, #fff);
```

```
  text-align: center;
  font-feature-settings: "clig" off, "liga" off;
  font-size: 14px;
  font-style: normal;
  font-weight: 500;
  line-height: normal;
  letter-spacing: 0.4px;
  border: 1px solid var(--Accent, #4723d9);
  margin-right: 16px;
}

.reset {
  display: inline-flex;
  padding: 10px 23px;
  align-items: center;
  gap: 10px;
  border-radius: 5px;
  border: 1px solid var(--7-border-secondary, #82868b);
  color: var(--1-theme-color-secondary, #82868b);
  text-align: center;
  font-feature-settings: "clig" off, "liga" off;
  font-size: 14px;
  font-style: normal;
  font-weight: 500;
  line-height: normal;
  letter-spacing: 0.4px;
  background-color: white;
}

.avatarcolumn2 p {
  color: var(--1-theme-color-body-text, #6e6b7b);
  font-feature-settings: "clig" off, "liga" off;
  font-size: 14px;
  font-style: normal;
  font-weight: 400;
  line-height: 21px; /* 150% */
}

.profileForm {
  display: flex;
  flex-direction: column;
  align-items: flex-start;
  gap: 24px;
  align-self: stretch;
  margin-left: 16px;
  margin-right: 16px;
  margin-bottom: 16px;
}
```

```css
.profileFormLine {
  display: flex;
  align-items: flex-start;
  gap: 24px;
  align-self: stretch;
}

.profileFormLine input {
  display: flex;
  width: 100%;
  height: 38px;
  padding: 8px 10px 8px 15px;
  justify-content: flex-end;
  align-items: center;
  border-radius: 5px;
  border: 1px solid var(--7-border-input, #d8d6de);
}

.profileFormLine input::placeholder {
  color: var(--1-theme-color-muted-placeholder-text, #b9b9c3);
  font-feature-settings: "clig" off, "liga" off;
  font-family: Montserrat;
  font-size: 12px;
  font-style: normal;
  font-weight: 400;
  line-height: 24px; /* 200% */
}

.profileFormLine input:focus {
  outline: none;
  border: 1px solid var(--1-theme-color-primary, #4723d9);
}

.profileDeletecontainer {
  display: flex;
  flex-direction: column;
  align-items: flex-start;
  align-self: stretch;
  border: 1px solid var(--gray-gray-2, #e9ecef);
  background: white;
  border-radius: 6px;
  box-shadow: 0px 4px 24px 0px rgba(0, 0, 0, 0.06);
  margin-bottom: 24px;
}

.profileDeleteHeader {
  display: flex;
  flex-direction: column;
  align-items: flex-start;
```

```
  justify-content: center;
  width: 100%;
  height: 60px;
  border-bottom: 1px solid var(--gray-gray-2, #e9ecef);
  background: white;
}

.profileDeleteHeader h4 {
  color: var(--1-theme-color-heading-display-text, #5e5873);
  font-feature-settings: "clig" off, "liga" off;
  font-size: 18px;
  font-style: normal;
  font-weight: 500;
  line-height: normal;
  margin-left: 23px;
}

.profileDeleteConfirmation {
  display: flex;
  flex-direction: column;
  width: 100%;
  background: var(--2-opacity-color-warning, rgba(255, 159, 67, 0.12));
}

.profileDeleteConfirmationRowOne {
  color: var(--1-theme-color-warning, #ff9f43);
  font-feature-settings: "clig" off, "liga" off;
  font-family: Montserrat;
  font-size: 14px;
  font-style: normal;
  font-weight: 800;
  height: 33px;
  align-items: center;
  justify-content: flex-start;
  display: flex;
}

.profileDeleteConfirmationRowTwo {
  color: var(--1-theme-color-warning, #ff9f43);
  font-feature-settings: "clig" off, "liga" off;
  font-family: Montserrat;
  font-size: 14px;
  font-style: normal;
  font-weight: 400;
  line-height: 21px;
  background: var(
    --5-alert-shadow-warning,
    linear-gradient(
      180deg,
```

```
      rgba(255, 159, 67, 0.16) 0%,
      rgba(255, 255, 255, 0) 94.05%
    )
  );
  height: 50px;
  align-items: center;
  justify-content: flex-start;
  display: flex;
  width: 100%;
}

.profileDeleteCheckbox {
  color: var(--1-theme-color-body-text, #6e6b7b);
  font-feature-settings: "clig" off, "liga" off;
  font-family: "Montserrat";
  font-size: 12px;
  font-style: normal;
  font-weight: 400;
  line-height: 18px; /* 150% */
}

.btnDelete {
  display: inline-flex;
  border-radius: 5px;
  background: var(--6-filled-button-danger-default, #ea5455);
  color: var(--1-theme-color-white, #fff);
  text-align: center;
  border: 1px solid var(--6-filled-button-danger-default, #ea5455);
  display: inline-flex;
  padding: 10px 22px;
  justify-content: center;
  align-items: center;
  gap: 8px;
  border-radius: 5px;
  text-align: center;
  font-feature-settings: "clig" off, "liga" off;
  font-size: 14px;
  font-style: normal;
  font-weight: 500;
  line-height: normal;
  letter-spacing: 0.4px;
}

.profileDeleteWarning {
  display: flex;
  flex-direction: column;
  align-items: flex-start;
  gap: 16px;
  width: 100%;
```

```
}

/*Para ecrãs pequenos (Telemoveis)*/
@media only screen and (max-width: 1400px) {
  .profileContainer {
    display: none;
  }
}
```

Anexos 1.6: Código de estilos CSS para a página Utilizadores

```
@font-face {
  font-family: Inter;
  src: url(../../assets/fonts/Inter-Regular.ttf);
}

.examsContainer {
  display: flex;
  padding: 40px;
  flex-direction: column;
  align-items: flex-start;
  gap: 16px;
  align-self: stretch;
  margin-left: 250px;
  transition: 330ms;
}

.examsContainerActive {
  display: flex;
  padding: 40px;
  flex-direction: column;
  align-items: flex-start;
  gap: 16px;
  align-self: stretch;
  transition: 550ms;
}

.headerRow {
  display: flex;
  justify-content: space-between;
  align-items: flex-start;
  width: 100%;
}

.headerRow h4 {
```

```
  color: var(--dark, #00162f);
  font-family: "Inter";
  font-size: 24px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
  letter-spacing: -0.72px;
}

.table {
  display: flex;
  padding: 24px;
  flex-direction: column;
  align-items: flex-start;
  align-self: stretch;
  border-radius: 12px;
  background: #fff;
  box-shadow: 0px 1px 1px 0px rgba(118, 135, 154, 0.24),
    0px 3px 8px 0px rgba(118, 135, 154, 0.08);
  overflow-x: auto;
}

.mainLine {
  display: flex;
  padding: 24px 16px;
  justify-content: space-between;
  align-items: flex-start;
  align-self: stretch;
  border-radius: 8px;
  background: var(--gray-gray-0, #f8f9fa);
  min-width: inherit; /* Inherit the minimum width from the parent */
}

.mainLine th {
  width: 548px;
  color: var(--gray-gray-6, #868e96);

  /* Labels/TH */
  font-family: "Inter";
  font-size: 13px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
  letter-spacing: 1.3px;
  text-transform: uppercase;
}

.dataLine {
  display: flex;
```

```
  padding: 16px;
  justify-content: space-between;
  align-items: center;
  align-self: stretch;
  min-width: inherit; /* Inherit the minimum width from the parent */
}

.dataLine th {
  width: 548px;
  color: var(--dark, #00162f);

  font-family: "Inter";
  font-size: 15px;
  font-style: normal;
  font-weight: 600;
  line-height: 150%;
  letter-spacing: -0.45px;
}

.finish_btn {
  display: flex;
  align-items: center;
  gap: 4px;
  color: var(--accent, #4723d9);
  background: none;
  border: none;

  font-family: "Inter";
  font-size: 15px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
}

.waiting {
  display: flex;
  align-items: center;
  gap: 4px;
  border-radius: 12px;

  color: var(--gray-gray-5, #adb5bd);
  font-family: "Inter";
  font-size: 15px;
  font-style: italic;
  font-weight: 600;
  line-height: normal;
}

.createExamButton {
```

```
  display: flex;
  align-items: center;
  gap: 8px;
  padding: 16px;
  border-radius: 12px;
  background: var(--accent, #4723d9);
  border: none;
  color: #fff;
  /* Labels/Button Label */
  font-family: Inter;
  font-size: 15px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
}
```

Anexos 1.7: Código de estilos CSS para a página de exame

```
@font-face {
  font-family: Inter;
  src: url(../../../assets/fonts/Inter-Regular.ttf);
}

.questionContainer {
  display: flex;
  padding: 40px;
  align-items: flex-start;
  align-content: flex-start;
  gap: 16px;
  align-self: stretch;
  flex-wrap: wrap;
  margin-left: 250px;
  transition: 550ms;
}

.questionContainerActive {
  display: flex;
  padding: 40px;
  align-items: flex-start;
  align-content: flex-start;
  gap: 16px;
  align-self: stretch;
  flex-wrap: wrap;
  transition: 350ms;
}

.headerRow {
```

```
  display: flex;
  align-items: center;
  justify-content: flex-start;
  gap: 16px;
  align-self: stretch;
  width: 100%;
}

@media (max-width: 768px) {
  .questionContainer {
    margin-bottom: 100px;
  }

  .questionContainerActive {
    /* on mobile add margin bottom because of previous and next buttons
*/
    margin-bottom: 100px;
  }

  .headerRow {
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    gap: 8px;
    padding: 40px 16px 16px 16px;
    align-self: stretch;
    width: 100%;
  }
  .arrowLeft {
    display: none;
  }

  .backIcon {
    display: none;
  }
}

.headerRow h4 {
  color: var(--dark, #00162f);
  font-family: "Inter";
  font-size: 24px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
  letter-spacing: -0.72px;
}

.questionNumber {
```

```
  flex: 1 0 0;
  color: var(--gray-gray-6, #868e96);
  font-family: "Inter";
  font-size: 15px;
  font-style: normal;
  font-weight: 400;
  line-height: 150%; /* 22.5px */
  letter-spacing: -0.45px;
}

.questionMainContainer {
  display: flex;
  align-items: flex-start;
  gap: 30px;
  align-self: stretch;
  width: 100%;
}

@media (max-width: 1000px) {
  .questionMainContainer {
    flex-direction: column;
    align-items: center;
    gap: 24px;
    align-self: stretch;
    width: 100%;
  }
}

.buttonsBackAndNext {
  display: flex;
  align-items: center;
  gap: 24px;
  align-self: stretch;
}

.buttonsBackAndNextMobile {
  display: none;
}

/* on mobile stick to the bottom and goes over all the screen */
@media screen and (max-width: 768px) {
  .buttonsBackAndNext {
    display: none;
  }

  .buttonsBackAndNextMobile {
    display: flex;
    justify-content: center;
    align-items: flex-end;
```

```
    gap: 8px;
    position: fixed;
    width: 100%;
    bottom: 0;
    background-color: #fff;
  }

  .bottomButtonsContainer {
    display: flex;
    padding: 0px 16px 24px 16px;
    justify-content: center;
    align-items: flex-end;
    gap: 8px;
    align-self: stretch;
    background-color: #fff;
    width: 100%;
  }
}

.previous_btn {
  display: flex;
  padding: 24px;
  justify-content: center;
  align-items: center;
  gap: 10px;
  flex: 1 0 0;
  border-radius: 12px;
  background: var(--gray-gray-1, #f1f3f5);
  border: none;
  color: var(--gray-gray-7, #495057);
  font-family: "Inter";
  font-size: 15px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
  width: 100%;
}

.next_btn {
  display: flex;
  padding: 24px;
  justify-content: center;
  align-items: center;
  gap: 10px;
  flex: 1 0 0;
  border-radius: 12px;
  background: var(--accent, #4723d9);
  border: none;
  width: 100%;
```

```
  color: #fff;
  font-family: "Inter";
  font-size: 15px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
}

.colorsColumn {
  display: flex;
  flex-direction: column;
  align-items: center;
  gap: 24px;
}

@media (max-width: 768px) {
  .colorsColumn {
    display: none;
  }
}

.questionContainer3 {
  display: flex;
  width: 100%;
  flex-direction: column;
  align-items: flex-start;
  gap: 30px;
  max-width: 400px;
}

@media (max-width: 1000px) {
  .questionContainer3 {
    display: flex;
    flex-direction: column;
    align-items: center;
    gap: 24px;
    margin-right: 0px;
    width: 100%;
    max-width: 100%;
  }
}

.loadingContainer {
  display: flex;
  flex-direction: column;
  align-items: center;
  gap: 24px;
  align-self: stretch;
  width: 100%;
```

```
}

/*Para ecrãs pequenos (Telemoveis)*/
@media only screen and (max-width: 1400px) {
  .questionContainer {
    display: none;
  }
}
```

Anexos 1.8: Código de estilos CSS para a página Login

```
@font-face {
  font-family: Inter;
  src: url(../../assets/fonts/Inter-Regular.ttf);
}

.login_container {
  display: flex;
  align-items: center;
  background: #fff;
  width: 100%;
  height: 100vh;
}

.column1 {
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  gap: 24px;
  width: 70%;
  height: 100%;
}

.column2 {
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  width: 100%;
  height: 100%;
}

.login_container_logo {
  display: none;
}
```

```css
/*Para ecrãs pequenos (Telemoveis)*/
@media only screen and (max-width: 1400px) {
  .login_container {
    display: flex;
    flex-direction: column;
    align-items: center;
    gap: 24px;
    width: 100%;
    justify-content: center;
  }

  .login_form_container {
    display: flex;
    width: 353px;
    padding: 0px 8px 16px 8px;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    gap: 32px;
    width: 100%;
    max-width: 500px;
  }

  .login_container_logo {
    display: flex;
    justify-content: center;
    align-items: center;
    gap: 8px;
    width: 100%;
  }

  .column2 {
    display: none;
  }
}
```

Anexos 1.9: Código de estilos CSS para a página Exames

```css
@font-face {
  font-family: Inter;
  src: url(../../assets/fonts/Inter-Regular.ttf);
}

.homeContainer {
  display: flex;
  padding: 40px;
  align-items: flex-start;
  align-content: flex-start;
  gap: 16px 482px;
  align-self: stretch;
  flex-wrap: wrap;
  transition: 550ms;
  margin-left: 250px;
  background: #f8f9fa;
}

.homeContainerActive {
  display: flex;
  padding: 40px;
  align-items: flex-start;
  align-content: flex-start;
  gap: 16px 482px;
  align-self: stretch;
  flex-wrap: wrap;
  transition: 350ms;
  background: #f8f9fa;
}

.headerRow {
  display: flex;
  justify-content: space-between;
  width: 100%;
  align-self: stretch;
}

.headerRow h4 {
  color: var(--dark, #00162f);
  font-family: "Inter";
  font-size: 24px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
```

```
  letter-spacing: -0.72px;
}

.seeall_btn {
  display: flex;
  align-items: center;
  color: var(--accent, #4723d9);
  border: none;
  background: transparent;
  font-family: "Inter";
  font-size: 15px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
}

.seeall_btn span {
  padding-right: 6px;
}

.cards {
  display: flex;
  flex-wrap: wrap; /* Allow cards to wrap to the next row on smaller
screens */
  gap: 20px; /* Adjust the gap between cards */
  justify-content: space-between; /* Distribute cards evenly along the
row */
}

.cardsmobile {
  display: none;
}

.card2 {
  display: flex;
  padding: 40px;
  max-width: 650px;
  flex-shrink: 0;
  border-radius: 12px;
  background: #fff;
  box-shadow: 0px 1px 1px 0px rgba(118, 135, 154, 0.24),
    0px 3px 8px 0px rgba(118, 135, 154, 0.08);
}

.card2_container1 {
  display: flex;
  flex-direction: column;
  align-items: flex-start;
  gap: 16px;
```

```
  margin-right: 16px;
}

.container_almosthere {
  display: flex;
  flex-direction: column;
  align-items: flex-start;
  gap: 8px;
  align-self: stretch;
  color: var(--gray-gray-6, #868e96);
  font-family: "Inter";
  font-size: 15px;
  font-style: normal;
  font-weight: 400;
  line-height: 150%; /* 22.5px */
  letter-spacing: -0.45px;
}

.container_almosthere h5 {
  color: var(--dark, #00162f);
  font-family: "Inter";
  font-size: 20px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
  letter-spacing: -0.6px;
}

.card2_container2 {
  display: flex;
  justify-content: center;
  align-items: center;
  width: 369.704px;
  height: 186px;
  flex-shrink: 0;
}

.text_percentage {
  color: var(--dark, #00162f);
  text-align: center;
  padding-bottom: 30px;

  font-family: "Inter";
  font-size: 36px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
  letter-spacing: -1.08px;
}
```

```css
.rectanglegreen {
  width: 48px;
  height: 48px;
  flex-shrink: 0;
  border-radius: 16px;
  background: var(--teal-teal-0, #e6fcf5);
  color: #20c997;
  display: flex;
  justify-content: center;
  align-items: center;
}

.finishexams_btn {
  display: flex;
  padding: 16px 24px;

  align-items: center;
  gap: 4px;
  border-radius: 12px;
  border: none;
  background: rgba(71, 35, 217, 0.05);

  color: var(--accent, #4723d9);
  font-family: Inter;
  font-size: 15px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
}

.card3 {
  display: flex;
  width: 100%;
  max-width: 369.704px;
  padding: 40px;
  align-items: flex-start;
  gap: 24px;
  flex-shrink: 0;
  border-radius: 12px;
  background: #fff;
  box-shadow: 0px 1px 1px 0px rgba(118, 135, 154, 0.24),
    0px 3px 8px 0px rgba(118, 135, 154, 0.08);
}

.card3_container {
  display: flex;
  flex-direction: column;
  align-items: flex-start;
```

```
  gap: 16px;
  width: 100%;
}

.card3_row {
  display: flex;
  width: 100%;
  flex-shrink: 0;
  flex-direction: column;
  overflow: hidden;
}

.card3_row h5 {
  color: var(--dark, #00162f);
  font-family: Inter;
  font-size: 20px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
  letter-spacing: -0.6px;
}

.card3_row h3 {
  color: var(--gray-gray-6, #868e96);
  /* Type/Regular */
  font-family: Inter;
  font-size: 15px;
  font-style: normal;
  font-weight: 400;
  line-height: 150%; /* 22.5px */
  letter-spacing: -0.45px;
  overflow: hidden;
  text-overflow: ellipsis;
  display: -webkit-box;
  -webkit-line-clamp: 3; /* Number of lines to show before truncating */
  -webkit-box-orient: vertical;
}

.rectangleyellow {
  width: 48px;
  height: 48px;
  flex-shrink: 0;
  border-radius: 16px;
  background: var(--yellow-yellow-0, #fff9db);
  color: #fab005;
  display: flex;
  justify-content: center;
  align-items: center;
}
```

```css
.seemyexams_btn {
  display: flex;
  align-items: center;
  gap: 10px;
  color: var(--accent, #4723d9);
  border: none;
  background: none;
  font-family: "Inter";
  font-size: 15px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
}

.carrousel {
  width: 100%;
  justify-content: center;
  align-items: center;
}

/*Para ecrãs pequenos (Telemoveis)*/
@media only screen and (max-width: 1400px) {
  .homeContainer {
    display: none;
  }

  .headerRow {
    display: flex;
    align-items: center;
    gap: 16px;
    width: 100%;
    align-self: stretch;
  }

  .headerRow h4 {
    color: var(--dark, #00162f);
    font-family: "Inter";
    font-size: 24px;
    font-style: normal;
    font-weight: 600;
    line-height: normal;
    letter-spacing: -0.72px;
  }

  .seeall_btn {
    display: flex;
    align-items: center;
    gap: 4px;
```

```
  border: none;
  font-family: "Inter";
  font-size: 15px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
}

.cards {
  display: none;
}

.cardsmobile {
  display: flex;
  flex-direction: column;
  width: 100%;
  max-width: 100%;
  gap: 24px;
}

.carrousel {
  width: 100%;
  justify-content: center;
  align-items: center;
}

.card2 {
  display: flex;
  flex-direction: column;
  align-items: flex-start;
  width: 100%;
  padding: 0px;
}
.card2_container1 {
  display: flex;
  flex-direction: column;
  align-items: flex-start;
  align-self: stretch;
  padding: 40px;
}

.card2_container2 {
  display: flex;
  justify-content: center;
  align-items: center;
  width: 100%;
  flex-shrink: 0;
}
```

```
  .card3 {
    padding: 0; /* Reduce padding for mobile */
    max-width: 100%;
  }

  .card3_container {
    display: flex;
    flex-direction: column;
    align-items: flex-start;
    gap: 16px;
    width: 100%;
    padding: 40px;
  }
}
```

Anexos 1.10: Código de estilos CSS para a página Painel

```
@font-face {
  font-family: Inter;
  src: url(../../assets/fonts/Inter-Regular.ttf);
}

.examsContainer {
  display: flex;
  padding: 40px;
  flex-direction: column;
  align-items: flex-start;
  gap: 16px;
  align-self: stretch;
  margin-left: 250px;
  transition: 330ms;
}

.examsContainerActive {
  display: flex;
  padding: 40px;
  flex-direction: column;
  align-items: flex-start;
  gap: 16px;
  align-self: stretch;
  transition: 550ms;
}

.headerRow {
  display: flex;
  justify-content: space-between;
  align-items: flex-start;
```

```
  width: 100%;
}

.headerRow h4 {
  color: var(--dark, #00162f);
  font-family: "Inter";
  font-size: 24px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
  letter-spacing: -0.72px;
}

.createExamButton {
  display: flex;
  align-items: center;
  gap: 8px;
  padding: 16px;
  border-radius: 12px;
  background: var(--accent, #4723d9);
  border: none;
  color: #fff;
  font-family: Inter;
  font-size: 15px;
  font-style: normal;
  font-weight: 600;
  line-height: normal;
}
```

Anexos 2: Análise de requisitos

Anexos 2.1: Descrição casos de uso - Login

Nome	**Login**
Descrição	Estes casos de uso têm como objetivo o utilizador realizar o login na aplicação
Pré-Condição	
Caminho Principal	1. O ator acede ao URL da aplicação. 2. O sistema apresenta a página de Login com os campos username e password; 3. O ator introduz o seu username e respetiva password. 4. O sistema envia toda a informação para a base de dados e autêntica o utilizador.
Caminho Secundário	4a) Erro ao de comunicação com o servidor; b) Erro Campos Vazios; c) Erro Email / Password estão incorretos.
Pós-Condição	O sistema apresenta a página de painel da aplicação.

Anexos 2.2: Descrição casos de uso – Criar questionário

Nome	**Criar Questionário**
Descrição	Estes casos de uso têm como objetivo o administrador criar um exame DISC.
Pré-Condição	Login Válido e ser Admin
Caminho Principal	1. O Administrador seleciona a página exames na Sidebar e seleciona opção de criar questionário; 2. O sistema apresenta um *container* para o Admin introduzir informações do exame; 3. O Administrador introduz os dados e seleciona a opção de criar exame; 4. O sistema envia a informação para a base de dados e apresenta o Admin na página de exame.
Caminho Secundário	4a) Erro ao de comunicação com o servidor; b) Erro Campos Vazios; c) Erro armazém já inserido.
Pós-Condição	O sistema envia alerta de "Exame criado com sucesso" se o pedido foi corretamente enviado.

Anexos 2.3: Descrição casos de uso – Criar utilizador

Nome	**Criar Utilizador**
Descrição	Estes casos de uso têm como objetivo o administrador criar um utilizador.
Pré-Condição	Login Válido e ser um dos seguintes utilizadores: • Admin • Staff Cliente • Cliente Administrador
Caminho Principal	1. O Administrador seleciona a página utilizadores na Sidebar e seleciona opção de criar utilizador; 2. O sistema apresenta um *container* para o Admin introduzir informações do utilizador; 3. O Administrador introduz os dados e seleciona a opção de criar utilizador; 4. O sistema envia a informação para a base de dados e apresenta o Admin o novo utilizador na tabela da página utilizadores.
Caminho Secundário	4a) Erro ao de comunicação com o servidor; b) Erro Campos Vazios; c) Erro utilizador já inserido.
Pós-Condição	

Anexos 2.4: Descrição casos de uso – Eliminar utilizador

Nome	**Eliminar Utilizador**
Descrição	Estes casos de uso têm como objetivo o administrador eliminar um utilizador.
Pré-Condição	Login Válido e ser um dos seguintes utilizadores: • Admin • Staff Cliente • Cliente Administrador
Caminho Principal	1. O Administrador seleciona a página utilizadores na Sidebar e seleciona opção de eliminar o utilizador; 2. O sistema elimina a informação da base de dados e apaga o utilizador da tabela de utilizadores.
Caminho Secundário	2a) Erro ao de comunicação com o servidor;
Pós-Condição	